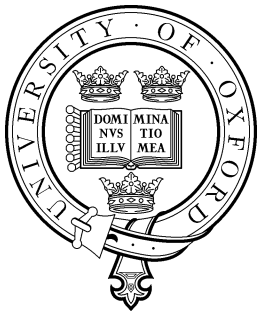
Accuracy Estimation for Sensor Networks



Hongkai Wen

Keble College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2013

Acknowledgements

I would like to thank my supervisor, Dr. Niki Trigoni, who has guided and supported me during my D.Phil study, especially when I repeatedly shifted my focus and explored new ideas. It is her insight, guidance and care that has made the last four years enjoyable and rewarding.

I also would like to give many thanks to the people in our group who have shared their valuable knowledge and experiences with me. I am very grateful of being a part of this team, and would thank all the members, both past and present, for helping me in so many ways.

Finally, I would like to thank the Engineering and Physical Sciences Research Council (EPSRC) for funding my research, and my family for their continued support throughout my study in Oxford.

Abstract

With sensor technology gaining maturity and becoming ubiquitous, we are experiencing an unprecedented wealth of sensor data. In most sensing scenarios, the measurements generated by sensor networks are noisy and usually annotated with some measure of uncertainty. The problem we address in this thesis is how to estimate the accuracy of the sensor systems based on the probabilistic measurements they provide. This problem is increasingly common in many settings, such as multiple sensing services are competing for the same group of users, detecting faults in large scale networks, or establishing trustworthiness of different individuals in social sensing. It is also challenging in many ways, for instance, the ground truth of the monitored states is absent, the users often lack a clear view of the implementation details of the sensor systems, and the reported accuracy can be misleading.

To address theses challenges, in this thesis we formulate the problem of estimating the accuracy of sensor systems in a general manner that applies to a broad spectrum of sensing scenarios. We then propose an accuracy estimation framework that breaks the problem into layers, which can be implemented in different ways. We present a novel inference-based accuracy estimation approach, which assesses the accuracy of sensor systems by comparing the reported measurements with the states inferred with the probabilistic measurements from all systems and available prior knowledge. We also propose a new learning-based approach for accuracy estimation, which employs novel parameter learning techniques. The learned parameters are either used to improve estimating the accuracy of sensor measurements, or to derive the accuracy of sensor systems directly in certain cases. We perform a systematic experimental evaluation on two datasets collected from real-world sensor deployments, where an array of different approaches are juxtaposed and compared extensively. We discuss how they trade accuracy for computation cost, and how this trade-off largely depends on the knowledge of the sensing scenarios. We also show that the proposed approaches outperform the competing ones in estimating accuracy and ranking the sensor systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Wireless sensor networks (WSNs) have become an active topic in both academic and industrial communities for many years. A wireless sensor network consists of a group of distributed nodes equipped with sensors, which cooperatively monitor physical or environmental conditions and forward the collected data via wireless links. As sensor technologies are gaining maturity, sensor networks have been widely used in many application scenarios that need surveillance, detection, monitoring and control [3]. For instance, active RFID tags [4] and low-frequency magnetic trackers [5] are used to record the activities of wild animals, and help understand their habits and social groups. Environmental sensor networks have been deployed in glaciers [6], volcanoes [7] and underwater environments [8], where data are collected for longer periods to facilitate studies in climate variability, ecological changes and pollution estimation. Sensor networks are also popular solutions for intelligent buildings and smart homes, where they are able to reduce energy consumption [9] and maintenance costs [10] by exploiting occupancy information of the residents. Equipped with sensors like EEG [11], ECG [12] and EMG [13], wearable body sensors networks are essential in many medical and healthcare applications [14, 15], and could potentially provide novel human-computer interfaces [16].

The above list is not exhaustive: in recent years, mobile devices such as smartphones and tablets have become powerful sensing platforms. Modern smartphones and tablets are programmable, and are shipped with a rich set of embedded sensors, such as accelerometer, gyroscope, magnetometer, camera, microphone, and GPS [17]. Those sensors allow the devices, which are typically carried around by a user, to acquire comprehensive information of the ambient environments, and therefore open up a broad variety of new sensing applications, such as social networks [18], smart transportation [19], activity recognition [20], and indoor positioning / navigation [21, 22]. One can envision that in the next few years,

the sensor systems will be ubiquitously integrated into our daily lives. In certain cases, they will need to coexist, collaborate and / or compete for users.

The problem that this thesis tries to address is **how to estimate the accuracy of the coexisting sensor systems, and leverage this information to assist decision making**. This is an important and timely problem in a number of different settings. Firstly, knowing how accurate the measurements of a sensor system are is paramount to deciding whether to use or pay for the service it offers. For example, if a positioning system consistently places a target at locations far away from the ground truth, the users should have a way of detecting the poor accuracy of this sensing service. Secondly, a user may be faced with the choice of selecting among multiple co-located sensor systems that offer a similar service (e.g. a WiFi-based vs. an FM-based indoor tracking [23] system in the same building). In this case, they should be in a position to compare or rank the accuracy of different systems. Thirdly, when a sensor system is first deployed, the administrator typically assumes a default noise model for the networked sensors. To detect when a sensor starts malfunctioning, it is critical to be able to assess when the accuracy of the measurements drops significantly below a certain threshold. Finally, the emergence of social sensing has raised the challenge of estimating the trustworthiness of human participants. When people report some observations (say, estimated air pollution levels), it is key to be able to assess the accuracy of the reported data.

Besides the accuracy of a sensor system, of course there are many other aspects that the users are interested in. For example, some sensor systems may impose a monetary cost on users while others could be offered for free; also the energy consumption of different sensor systems can vary significantly; and some users may be more sensitive about their privacy and thus prefer privacy-preserving systems. This thesis assumes that this information is readily available from other sources, e.g. web services or monitoring tools, and focuses on estimating the accuracy. We claim that this is a key step towards empowering the users to choose the sensor systems that can achieve their desired trade-offs between different quality metrics.

1.2 Research Challenges

We have identified the following main challenges in addressing the accuracy estimation problem from our exercises in real sensing applications:

- **Ground truth of the measured signals is absent:** - Sensor systems report measurements of physical signals, such as the positions of a user, or the light intensity of a room. If the true values of the measured signals were known, evaluating their accuracy would be trivial: conceptually, we can always compare a sensor measurement with the ground truth, and reason about its accuracy based on how different it is from the true value, given a metric. Unfortunately in most sensing scenarios, it is either impossible or impractical to obtain the ground truth, and thus the accuracy cannot be measured directly.

- **Sensor systems are black boxes:** - With their ever-increasing capabilities and ubiquity, many sensor systems have become general purpose sensing platforms, which can be accessed by a large number of different users [1]. Due to security and privacy issues, sensor systems may only provide the users with interfaces for data acquisition, but hide most of the implementation and deployment details, such as sensor locations or sensor types, which can be clues to understand the accuracy of generated measurements. For instance, a WiFi-based positioning system with a dense network of access points (APs) is likely to be more accurate than a system with only a few APs. In the absence of such information, sensor systems present themselves as *black boxes*, and users only subscribe to the sensing services they provide, with little or no knowledge about how they work. They often lack a clear view of the implementation details of the sensor systems.

- **Sensor measurements are uncertain:** - In many sensing scenarios, sensor measurements are *noisy* and prone to error. In practice, they are typically annotated with some measure of uncertainty, e.g. a confidence interval or region. For instance, a sensor system that monitors the temperature of a room would report a temperature range rather than a single value, e.g. $20 \pm 3°$C; and similarly a positioning system may associate the estimated user position with an error ellipse, indicating that the user has a high probability (e.g. 95%) of being within this ellipse at a given timestamp. In general, sensor measurements can be represented as probability distributions over the domain of measured signals, and they are hereafter referred to as *probabilistic measurements*. Those probabilistic measurements are typically harder to manage and process than the deterministic ones, and therefore evaluating their accuracy requires more sophisticated approaches.

- **Reported accuracy information can be misleading:** - For a probabilistic measurement, the annotated uncertainty information, i.e. the error ellipsoid, represents the

reported belief on accuracy. For example, when a positioning system pairs the estimated position of a user with an error ellipse, a smaller error ellipse means that the system thinks this measurement is more accurate, and vice versa. However, this reported accuracy information is merely a belief provided by the sensor system, and can often be misleading. Sensor systems can be either too optimistic or too pessimistic about error in their measurements, and the reported accuracy is not always a reliable indicator of the real accuracy. Therefore, we cannot simply rely on the reported uncertainty information to reason about the real accuracy of the sensor systems.

- **The accuracy of a sensor system is context-dependent:** - In practice, the accuracy of a sensor system may vary in different contexts, such as space or time. For example, consider an indoor positioning system that uses WiFi signal strength to localise mobile phones carried by the users. Firstly, the accuracy of this sensor system can vary significantly across space, e.g. it should be more accurate in open spaces than in cluttered environments, where most RSSI readings are non-line-of-sight (NLOS). Secondly, it may work better during the night than at daytime since there is less wireless signal interference. Finally, the system may perform differently for different devices (users) simply because of the heterogeneous WiFi hardware. Therefore it is not reliable to infer the accuracy from previous experiences when the context is different; at the same time, training before every use is too labour-intensive and impractical.

1.3 Illustrative Examples

The problem of accuracy estimation studied in this thesis is important in many scenarios, and the proposed accuracy estimation techniques can be used in a broad range of sensing applications:

- **Sensor system selection:**- For example, it is very likely that in the near future multiple sensor systems will be co-located and monitor the same physical signals, such as the environmental variables, traffic volume, trajectories of pedestrians, etc. In those cases, the users may need to select one or a few systems to use (due to the limited budget on cost, energy or network bandwidth), to acquire the information with desired accuracy. Therefore, knowing the accuracy of measurements reported by coexisting sensor systems is vital for the users to make informed decisions as to which system to task.

- **Fault detection:**- Another example could be fault detection in large scale sensor networks. In most applications, measurements generated by sensors are considered to be *noisy*. When firstly deployed, the sensors are usually assumed to have certain default noise models. However, due to changes in the environment, the actual noise profiles of individual sensors can vary significantly. Moreover, in some applications such as social sensing where human participants are reporting measurements, it is not even possible to have calibrated noise models a priori. For those cases, assessing the accuracy of the sensors / human participants is the key to determine their reliability and trustworthiness, and therefore can be used to detect when and where the sensor network is malfunctioning (e.g. when the measurement accuracy of a node drops below a certain threshold).

- **Sensor fusion:**- Finally, understanding the accuracy of sensor measurements is an important prerequisite for many sensor fusion applications. For instance, autonomous robotic systems (e.g. self-driving vehicles) are typically equipped with multiple types of sensors, such as cameras, laser scanners, etc. The sensory data is fused to derive the view of the world (e.g. the map), which is then used to localise and navigate the robots through the workspace. In practice, the sensors can behave very differently, e.g. cameras are prone to low lighting / lens flare, and lasers suffer from ground strike and wet / frozen conditions. In this case, estimating the sensor accuracy is paramount: with the accuracy information, the robots can selectively fuse the sensor measurements to obtain a better knowledge of the environment in different context (e.g. by assigning different weights according to the sensor accuracy).

Of course there are many other applications that could benefit from the accuracy estimation techniques introduced in this thesis. For simplicity, in the following we consider two real sensing scenarios as the running examples: a) an indoor positioning scenario, and b) an environmental monitoring scenario. The two scenarios will be used extensively to elaborate the proposed accuracy estimation approaches throughout this thesis and in the experimental evaluation of the proposed approaches.

1.3.1 An Indoor Positioning Scenario

Indoor positioning has become an increasingly active research topic for both industry and academia. Recently, there is an explosion of indoor positioning techniques that compete for adoption in the global smart device market, including WiFi / Bluetooth / Zigbee based localisation, RFID tracking, visible light communications, indoor GPS, localisation with

photo-acoustic signatures, FM radio signals, magnetic fields, etc. In this climate, it is reasonable to expect that many large indoor venues, such as shopping centres or airports, will soon be outfitted with multiple indoor positioning systems, which are coexisting and providing positioning services.

In this emerging ecosystem, the accuracy of the coexisting systems can vary for different users, in different areas of the space, or even in different time periods, and it is not clear which positioning system can provide the best services throughout. On the other hand, users are likely to prefer the most accurate system, and thus it is crucial for them to understand the accuracy of the systems that are available. With this knowledge, the users will be able to make informed decisions as to which positioning system to task when moving across the indoor space, and when to switch from one system to another. For instance, a user may choose the most accurate positioning system available to use when she first enters the building. As she moves around, she may prefer to switch to another system which is more accurate in the area that she is currently visiting.

Estimating the accuracy of an indoor positioning system is a challenging task. The actual trajectory of the user is unknown, so the accuracy of the system cannot be evaluated directly. Also we normally have no idea about which localisation modalities or algorithms are used by the systems, or where the sensing infrastructure is deployed. The only information we possess is the reported location measurements. Typically, a position measurement consists of an estimated user position and the associated error ellipse. As stated above, the error ellipse represents the reported uncertainty in the measurement. However, this accuracy may not be a faithful estimation of the real accuracy. More specifically, we have observed the two distinct cases in real indoor positioning systems where this happens:

- *The actual locations of a user consistently falls out of the reported error ellipses.* This can happen when the localisation algorithm used by the system fails to cope with changes of the environment. For instance, in Figure 1.1(a), a positioning system that is based on WiFi RSSI triangulation performs well in the straight corridor, while when the user is moving towards the corner, the system consistently underestimates the error in its measurements. Another example may happen in safety-critical environments, where one or more positioning systems may be compromised by the adversary and made to consistently report wrong locations with very high reported accuracy.

- *The size of the error ellipses reported by different positioning systems can lead to wrong conclusions about which system is more accurate.* For example, consider the two estimated trajectories of a single user shown in Figure 1.1(b). The top trajectory

(a) (b)

Figure 1.1: (a) At the bottom left corner, the true locations fall out of the reported error ellipses. (b) The reported error ellipses are misleading as to which is the more accurate positioning system.

is provided by a positioning system using inertial sensors and the bottom one is from the commercial WiFiSLAM system [21]. One can immediately see that the estimated trajectory on top is far better than the bottom one, but the reported accuracy is significantly lower. Note that in both trajectories, the real positions of the user are covered by the reported error ellipses.

In this sensing scenario, the aim is to provide a solution which is able to estimate the accuracy of the coexisting positioning systems and coordinate access to the systems accordingly.

1.3.2 An Environmental Monitoring Scenario

The second example considered in this thesis concerns an indoor environmental monitoring scenario. For instance consider a smart building application, where certain physical signals such as temperature, humidity or light intensity, are required by multiple parties: e.g. the building manager may use the temperature or light intensity values to infer the energy consumption, while a company residing in the building may analyse office occupancy with similar information. In this case, different parties may deploy their own sensor networks to monitor the surrounding environment. Those systems can be highly overlapping, and therefore need to be shared so that users can task the suitable systems to collect useful information.

One possible approach of sharing the sensing services provided by different sensor systems is to create a set of unified APIs, which allow the users to query the coexisting systems,

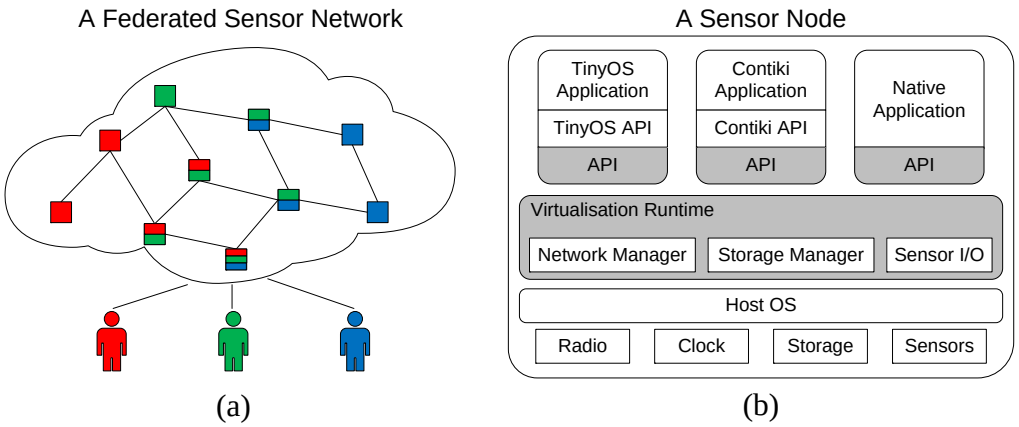


Figure 1.2: (a) An example of a federated sensor network. The network accommodates three virtual sensor systems, which are tasked by three different parties respectively. (b) The architecture of a node in federated sensor networks, which can host multiple sensing applications (taken from the SenShare system [1]).

e.g. as in the SenseWeb system [24]. Another approach which has recently attracted a lot of interest, is to deploy shared sensing infrastructure that can host multiple sensing applications. The SenShare system [25, 1] considers an indoor setting, and proposes the concept of *federated sensor networks*, which allow sensing infrastructure to be shared among multiple virtual sensor systems. The virtual systems are implemented on top of the real sensing hardware (i.e. the sensor nodes) by virtualisation, and each of them is operated individually. Therefore in federated sensor networks, the virtual sensor systems and real sensing infrastructure are decoupled, in the sense that one virtual sensor system can span over an arbitrary number of sensor nodes, while a node can host multiple virtual sensor systems. Figure 1.2(a) shows an example of a federated sensor network, and Figure 1.2(b) illustrates the architecture of a node in the federated network (taken from the SenShare system [1]).

Understanding the accuracy of the coexisting sensing systems is essential in the context of environmental monitoring. Firstly, it is likely that some sensing systems are monitoring the same variables, and users would like to choose the one with the highest accuracy. For instance in the smart building scenario, if multiple sensor systems are offering information about the temperature of an area, the users would want to use the most accurate one. Secondly, some systems may become faulty or inaccurate due to changes in the environment, and it is important for the users to realise this and avoid using them. Thirdly, some malicious systems may report inappropriate data on purpose, e.g. generating sensor measurements with unreasonably high reported accuracy to attract adoption, and it is crucial that the administrator can discover and isolate those systems.

However, assessing the accuracy of the coexisting sensor systems in this scenario is

not trivial. Firstly, the true values of the monitored variables, e.g. the temperature, are usually not known, and thus the accuracy of measurements reported by different sensor systems can not be evaluated directly. Secondly, we typically possess little or no knowledge on how the shared sensor systems work: they only provide APIs or they are running on virtualised infrastructure, therefore it is hard to estimate how optimistic / pessimistic they are in estimating uncertainty of their measurements. Finally, as in the previous example, the reported accuracy generated by those sensor systems, e.g. the confidence intervals of the measured values, may not be faithful representations of the real accuracy.

To sum up, in the context of environmental monitoring, it is challenging, yet important, to fairly assess the accuracy of the coexisting sensor systems, and provide the users with such information to enable decision making.

1.4 Problem Definition

Now that we have motivated the need for accuracy estimation through two illustrative scenarios, we are in a position to introduce our model, explain key assumptions, and formulate the accuracy estimation problem in the context of sensor networks.

1.4.1 Model and Assumptions

1.4.1.1 Monitored states

Let x_t be the real value of the signal that a sensor system is measuring at a given timestamp t. For example, x_t could be the temperature of a room, or the location of a user at time t. We assume time is discrete and finite, i.e. the timestamps $t = 1 : T$ are a totally ordered set. In the following text, we refer to x_t as the *state*, and denote the value domain of x_t with a set Ω. Without loss of generality, this thesis focuses on dynamic processes, where the monitored state evolves over time. Stationary processes can be viewed as special cases with only one timestamp, i.e. $T = 1$, where the measurements collected during the entire period are all collapsed to $t = 1$.

1.4.1.2 Sensor systems

We consider the general case that M coexisting sensor network systems $sn_1, ..., sn_M$ are monitoring the underlying states x_t. They provide sensing services by reporting streams of sensor measurements. Let z_t^m be the measurement generated by the m-th sensor system at time t, $1 \leq m \leq M$, $1 \leq t \leq T$. Depending on the sensor system, z_t^m could be either

a single estimated value, or a value paired with certain error bounds. This thesis assumes that the measurements are *probabilistic*, i.e. z_t^m is a random variable defined on Ω with the observed probability distribution $p(z_t^m)$. Deterministic measurements can be viewed as special cases where $p(z_t^m)$ is reduced to a point distribution.

1.4.1.3 Sensing applications

A sensing application is the agent that requires information on the monitored states x_t. It typically selects the sensor systems to task based on its accuracy requirements. We assume that a sensor application describes its accuracy requirements by providing a pair $\langle f_\epsilon, \boldsymbol{A} \rangle$. The first element f_ϵ is an *accuracy metric*, which is a function that maps any given sensor measurement z_t^m to its accuracy. The second element $\boldsymbol{A}$ is a set of attributes that specifies the *accuracy index* on which the sensing application would like to aggregate the accuracy of measurements.

Accuracy metric: Theoretically, the accuracy metric f_ϵ provided by the sensing application can be any function that takes the measurement z_t^m as input and computes the accuracy of z_t^m. For simplicity, this thesis assumes that the accuracy of a sensor measurement z_t^m is a *scalar value*, and it is evaluated with respect to the ground truth x_t (or an estimate of the ground truth). Therefore, the accuracy of a sensor measurement z_t^m is the output of the accuracy function $f_\epsilon(z_t^m; x_t)$. Chapter 3 will present two accuracy metrics, and show their viability in different sensing scenarios.

Accuracy index: The accuracy index is a multidimensional array, where each element is the aggregated accuracy of multiple sensor measurements. An accuracy index is specified by the set of attributes $\boldsymbol{A}$ provided by the sensing application, such as time, location, or user ID. Each attribute $A \in \boldsymbol{A}$ uniquely determines one dimension of the accuracy index. The value assignments of the attributes $\boldsymbol{A}$ are used to group the sensor measurements and aggregate their accuracy. For instance, the accuracy of a temperature-monitoring sensor system in room 100 depends on the aggregated accuracy of all the temperature measurements attached to room 100, and the collection of this accuracy for all rooms forms the accuracy index over the attribute *location*. Chapter 3 will explain the process of building the accuracy index for the sensor systems in more detail.

1.4.1.4 Prior state distribution

We assume that in certain timestamps, there may be certain prior knowledge available on how the state x_t is distributed. We refer to such information as the *prior state distributions*, or simply *priors*, and use a random variable ρ_t to represent this knowledge on the state at timestamp t. Consider an indoor positioning scenario where the states are the actual

locations of the user. Planned events such as calendar entries, or social interactions like store check-ins, may directly reveal the user location at the given timestamp [26]. For instance, if the calendar of a user shows that she will be having a meeting in Room 100 at 3pm, then it is very likely that her real location (the state) at 3pm is in Room 100. Chapters 4 and 5 will explain how to incorporate the available prior state distributions into the accuracy estimation process.

1.4.1.5 Estimated states

In practice, the real state x_t is not known or measured exactly, but needs to be estimated. We therefore denote the estimated states as a random variable $\hat{x}_t$ defined on Ω with a probability distribution $p(\hat{x}_t)$. In our context, $\hat{x}_t$ is evaluated with the observed sensor measurements and the available priors, and it represents the best guess of the real state x_t. We will show in later chapters that the estimated state $\hat{x}_t$ can have significant impact on the quality of accuracy estimation.

1.4.2 The Accuracy Estimation Problem

As discussed above, the sensing application needs to sense the physical phenomenon x_t. M coexisting sensor systems offer sensing services by providing probabilistic sensor measurements z_t^m, $1 \leq m \leq M$, $1 \leq t \leq T$. The ***accuracy estimation problem*** studied in this thesis is to assess the accuracy of the M sensor systems according to the accuracy requirements $\langle f_\epsilon, \boldsymbol{A} \rangle$ from the sensing application, given all the observed sequences of sensor measurements $z_{1:T}^1, ..., z_{1:T}^M$ and the prior knowledge on the monitored states. This will empower the sensing application to proactively select the systems that offer the best performance in each context according to $\langle f_\epsilon, \boldsymbol{A} \rangle$.

1.5 Contributions

To enable the sensing applications to easily assess the accuracy of different sensor systems and leverage this knowledge to make informed decisions, this thesis presents a class of novel accuracy estimation approaches for sensor systems, and complements that with a comprehensive evaluation of them. Concretely, the technical contributions of this thesis are as follows:

1. We have motivated the accuracy estimation problem in a wide range of real-world sensing scenarios. For instance, in the context of pricing sensing services, understanding their accuracy if they are competing for the same group of users, detecting faults in large scale networks, and establishing trustworthiness of different individuals in social sensing. We have identified the important challenges in addressing this problem, namely a) the ground truth is absent; b) the sensor systems are black-boxes; c) the sensor measurements are uncertain; d) the reported accuracy can be misleading; and d) the accuracy of sensor systems is context-dependent. We have formulated the important concepts and assumptions, and defined the accuracy estimation problem in a general manner.

2. We propose a general framework to address the accuracy estimation problem, which contains four layers: pre-processing, state estimation, accuracy estimation and accuracy indexing. For state estimation, we create a taxonomy of approaches, ranging from simple voting schemes, to inference- and learning-based techniques. For accuracy estimation, we propose two metrics, one based on *proximity* and one on *similarity* between the probabilistic measurements and the state estimates. Those metrics evaluate the accuracy of a measurement from different perspectives, and are thus suitable for different classes of applications. For accuracy indexing, we propose an accuracy indexing scheme that can build accuracy indices of sensor systems by aggregating and interpolating the accuracy of measurements over given attributes.

3. Following the proposed framework, we develop a novel inference-based approach to estimating the accuracy of different sensor systems. We model the monitored signal as a stochastic process with latent states, which accepts the probabilistic sensor measurements as observations, and takes the prior state distributions into account. We show how to firstly estimate the latent states given the available measurements, and then use the estimated states to evaluate the accuracy of the sensor measurements. We present two cases of this approach, the *static* and the *dynamic*. The static inference approach ignores any temporal correlations between the latent states, while the dynamic inference considers the dynamics of the monitored process and uses previous and future observations to help approximate the current state.

4. We propose a new learning-based approach for the accuracy estimation problem, which has two variants. The first variant shares the same formulation with the inference-based approach, but uses learning techniques to re-estimate the model parameter before inferring state. The second variant is suitable for the cases where the

accuracy needs to be indexed over the monitored states, where the inference-based approach fails to work. We show that the proposed approach can estimate accuracy by jointly addressing the problems of accuracy estimation and indexing. For both the two variants, we propose novel parameter learning approaches based on the Expectation Maximisation (EM) scheme, which are able to learn the model parameters that are the most consistent with the probabilistic sensor measurements and the prior state distributions.

5. We perform a systematic experimental evaluation of the proposed accuracy estimation approaches in the context of two real-world sensing scenarios. We build a research testbed of indoor positioning, where we implement and deploy a variety of indoor positioning techniques with different localisation modalities. We collect the positioning data in two indoor environments for more than 20 days. We also consider the widely used Intel Lab dataset [27], which contains environmental sensor readings of a lab environment including temperature, humidity and light intensity for about two months. We evaluate the proposed approaches on the two real sensor datasets, and show that our approaches outperform the baseline and competing approaches in both scenarios. We also perform a thorough comparison study of the proposed accuracy estimation techniques, in terms of accuracy estimation quality, computational cost, sensitivity to the prior state distributions and coexisting sensor systems, etc. We show that different approaches have their own merits in different contexts, and by carefully selecting and tuning them, we can achieve the desired trade-offs in different scenarios.

1.6 Publications

The main contributions of this thesis have already been published at the following international conferences:

1. Hongkai Wen, Zhuoling Xiao, Andrew Symington, Andrew Markham and Niki Trigoni. "Comparison of Accuracy Estimation Approaches for Sensor Networks". In *Proceedings of IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)* , 2013.
   This paper presents a taxonomy of techniques for accuracy estimation in the context of multiple coexisting sensor systems and compares inference- with learning-based techniques.

2. Hongkai Wen, Zhuoling Xiao, Niki Trigoni and Phil Blunsom. "On Assessing the Accuracy of Indoor Positioning Systems". In *Proceedings of European Conference on Wireless Sensor Networks (EWSN)*, 2013. (Best paper award)
   This paper proposes a novel learning algorithm, which is a variant of the Baum-Welch algorithm for estimating the accuracy of co-located indoor positioning systems.

The work in this thesis also contributes to the following published papers:

1. Zhuoling Xiao, Hongkai Wen, Andrew Markham, Niki Trigoni, Phil Blunsom and Jeff Frolik. "Identification and Mitigation of Non-line-of-sight conditions Using Received Signal Strength". In *Proceedings of IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)* , 2013.
   This paper proposes novel non-line-of-sight (NLOS) identification and mitigation approaches for WiFi received signal strength measurements. The experiments of this work are performed on the testbeds built by this thesis, where the indoor positioning systems created in this thesis are used.

2. Dan Olteanu, Hongkai Wen. "Ranking Query Answers in Probabilistic Databases: Complexity and Efficient Algorithms". In *Proceedings of IEEE International Conference on Data Engineering (ICDE)*, 2012.
   This paper proposes efficient techniques to rank query answers in probabilistic databases. The paper is motivated by the idea of using prior knowledge to check the accuracy of sensor streams discussed in this thesis. However, a different approach is taken to check for compliance with prior information, inspired by work on query processing for probabilistic databases.

1.7 Thesis Structure

The rest of this thesis is organised as follows. Chapter 2 provides an overview of related work. The following three chapters present our proposed approaches. Chapter 3 presents a general framework for accuracy estimation consisting of four layers: 1) pre-processing, 2) state estimation, 3) accuracy estimation and 4) accuracy indexing. Chapter 4 focuses on the state estimation layer and proposes and inference-based approach to tackling this problem. Chapter 5 presents a learning-based approach to state and accuracy estimation (layer 2 and 3). Chapter 6 presents the experimental evaluation of the proposed approaches

in real-world sensing scenarios, and provides a comprehensive discussion of the experiment results. Finally, Chapter 7 concludes this thesis and outlines areas for future work.

Chapter 2

Background

The work proposed in this thesis draws heavily from two areas of research: *state estimation* and *parameter estimation*. Concretely, the proposed inference-based approach for accuracy estimation falls into the general class of state estimation approaches in sensor networks, and borrows a lot of ideas from the work on probabilistic inference approaches. The learning-based accuracy estimation approach is naturally related to the research on parameter estimation in sensor networks, and is also connected to the learning problems studied in graphical models. Finally, the work in this thesis is also related to a range of other research area, including fact finding in information networks, cost-constrained data acquisition in sensor networks and recommendation systems.

This chapter provides background information and surveys existing work in the related research areas. The remainder of this chapter is organised as follows: Section 2.1 briefly explains the related probabilistic inference techniques in machine learning, and describes the existing sensor network research on statistical inference. Section 2.2 reviews the learning techniques used in graphical models, and discusses the current work on parameter estimation in sensor networks. Each of the sections is concluded with a discussion of how the proposed approaches are related to existing work. Finally, Section 2.3 discusses a few other existing research efforts that have close connections with this thesis.

2.1 Inference and State Estimation

The inference-based accuracy estimation approach proposed in this thesis is closely related to research in inference and state estimation. In the proposed approach, the monitored signals (which are unknown) and the sensor measurements (reported by the systems) are modelled with probabilistic graphical models (PGMs). The approach infers the monitored

signals with the measurements from all the sensor systems, and then estimates the accuracy of the measurements with respect to the estimated signals. Therefore in the proposed inference-based approach, the key step for accuracy estimation is to estimate the latent states given all the sensor observations, which is directly related to the inference and state estimation techniques surveyed in the rest of this section. In the following, we first explain the related inference and state estimation techniques in general graphical models, then describe their applications in sensor networks, and finally discuss how our approach is connected to these existing techniques.

2.1.1 Inference and State Estimation in Graphical Models

Inference is one of the fundamental problems in machine learning and artificial intelligence research, especially for graphical models [28, 29, 30]. A *probabilistic graphical model* (PGMs) contains a set of *nodes* connected by *edges*, where each node represents a random variable, and the edges denote the probabilistic relationships between the variables. Depending on whether the edges are directed, the PGMs can be divided into two types: *Bayesian networks* (directed graphs) [29] and *Markov networks*, or *Markov random fields* (undirected graphs) [31]. The former are widely used in areas such as signal processing [32], object tracking [33], and data mining [34], while the latter are popular tools for image reconstruction [35], segmentation [36] and information retrieval [37].

In our context, the physical signal we would like to monitor is typically unknown, i.e. *latent*, and can only be measured through a series of sensor observations. Therefore, this thesis is particularly related to one special type of graphical models, the *state space models* (SSMs), or *latent process models* [38]. The state space models contain two sets of random variables, the *latent states* and the *evidences*, and assume that state variables follow a stochastic process, which emits the evidence variables at different timestamps. The state space models have been widely used in many areas, such as speech recognition [2] and target tracking [19].

As in general graphical models, the process of computing the posterior distributions of the latent variables given the values of observed variables is called *probabilistic inference* [29]. In practice, the basic inference tasks for state space models are as follows [29]:

- *Filtering*: Given all the observations to date, filtering computes the posterior distribution of the *current* state;

- *Prediction*: Given all the observations to date, prediction computes the distribution of a *future* state;

- *Smoothing*: Given all the observations to date, smoothing computes the posterior distribution of a *past* state;

- *Viterbi decoding*: Given a sequence of observations, the Viterbi decoding computes the state sequence that is most likely to have generated the observation sequence.

Among those tasks, filtering and smoothing are closely related to the work in this thesis. For ease of exposition, in the following text we consider the two well studied state space models: the hidden Markov models (HMMs) and linear dynamical systems with Gaussian variables (LDSs). HMMs are the simplest form of dynamic Bayesian networks, which assume the state space is discrete. LDSs are dynamical systems (whose state space is continuous) with linear evaluation functions, which have closed form solutions for filtering and smoothing. In the following, we briefly describe the existing filtering and smoothing algorithms in the context of HMMs and LDSs.

In general, filtering is also referred to as *state estimation*. The goal of filtering, or state estimation is to compute the posterior distribution of the current state (latent variables) given all the previously observed evidence. The filtering problem of dynamic models can be addressed with both *exact* and *approximate* algorithms. For HMMs, the most widely used exact filtering algorithm is the forward algorithm [2], which is based on dynamic programming techniques. For LDSs, the *de factor* algorithm for exact filtering is the Kalman filter [38], which takes the noisy observations (corrupted by white noise) as input, and produces optimal state estimates. For more complex models where no exact filtering algorithm is available (e.g. nonlinear models), the filtering problem can be addressed by approximate algorithms, such as the sequential Monte Carlo (SMC) algorithms [39] that use a large number of particles to approximate the state distribution, or the extended Kalman filter (EKF) that relies on linearisation, and so on.

Unlike filtering which estimates the current state based on previously observed evidence, the smoothing problem estimates the posterior state distribution given all the previous and future observations. Therefore typically the smoothing problem in state space models (SSMs) can be addressed by general inference techniques in probabilistic graphical models (PGMs). For instance, the exact smoothing algorithms include the variable elimination algorithm [40], belief propagation [41], junction trees [42] and recursive conditioning [43, 44]. Particularly for HMMs and LDSs, the most popular exact smoothing algorithms are the forward-backward algorithm [2] and the Kalman smoother [45] respectively. As in filtering, for general SSMs which are not subject to an exact smoother, the problem has to be addressed with approximate algorithms. One of the most widely used

approximate smoothing algorithms is the particle smoother [46], which however, can become prohibitively expensive when the dimension of the state space is high. There are also methods that do not suffer from the curse of dimensionality, such as the variational methods (also known as variational Bayes) [47, 48], which approximate the posterior state distribution with a tractable distribution that is carefully selected to minimise the difference between the real and approximated posterior state distributions.

2.1.2 Inference and State Estimation in Sensor Networks

A large body of research in sensor networks has applied statistical inference to monitoring tasks. In this context, the monitored physical phenomenon, e.g. the temperature, humidity, light intensity or pollution levels, is typically assumed to vary over time. The measurements reported by the sensor network provide a noisy view of the instrumented environment. The inference problem for sensor networks is to reason about the unknown states given the noisy sensor measurements. It has been defined in various forms for different sensing applications such as sensor calibration [49], target tracking [50], data modelling and contour finding [51].

Inferring absent sensor readings: A common approach to address this inference problem is to use techniques such as Kriging [52] and Gaussian Processes (GPs) [53] to interpolate between sensor readings and infer the values of the monitored environmental variables. These techniques take into account the spatial and temporal correlations in sensor readings, and are able to incorporate the noise encoded in the sensor measurements. Concretely, they use customized non-linear Gaussian processes to estimate the monitored states over time and space, particularly in places where there is no sensor, or when sensors have failed or simply do not generate readings at certain timestamps. For example, the work in [54] uses Gaussian process regression to interpolate the sensor measurements, and combine information from different types of observations to estimate the underlying states. Osborne et al. have proposed a computationally efficient implementation of GPs for sensor network applications in the context of environmental sensing [55]. In that work, they use a multi-output Gaussian process to process the observed environmental data in real-time with minimal domain knowledge, where the hyper-parameters of the GPs are estimated by a Bayesian Monte Carlo scheme.

Estimating monitored signals: Besides inferring the latent states by interpolation and regression, a lot of work has also considered using probabilistic models to estimate the monitored signals. In those scenarios, the state space models are often used to represent the dependencies between the latent states and the sensor measurements. A well-studied

example is that of node tracking, where the physical locations of moving objects are tracked by fixed and / or mobile sensors, and the task is to estimate real location of the objects (the states) given the noisy sensor measurements (the observations). Hidden Markov models (HMMs) are commonly used in this context. A large body of work has investigated HMMs for map matching, which is the problem of finding the most likely trajectory that accounts for measurement noise and known map constraints [56, 57, 19, 58, 59]. More specifically, VTrack uses mobile phones mounted in cars to estimate road travel times using a sequence of inaccurate position observations [19]. EasyTracker uses HMMs in the context of transit tracking, and uses the inferred tracks to detect transit stops and predict arrival times [59]. An HMM-based approach is also used in CTrack, where the goal is to associate a sequence of cellular fingerprints to a sequence of road segments on a known map [60].

Sensor tracking: In addition, Bayesian estimation techniques such as Kalman and particle filters have also been broadly used for state estimation / inference problems in the area of localisation / tracking. For example, HMMs-based localisation systems have been widely used for both outdoor and indoor localisation [61, 62, 58]. The work in [62] uses the first-order HMMs to track the users, where latent states represent the user locations. It fuses the WiFi fingerprinting information with inertial measurements to infer the trajectories of the user. The AutoWitness system [61] however, uses the second-order HMMs where the transitions between pairs of locations are modelled as states. The work in [63] considers the continuous cases where the location of a user is represented as a continuous random variable. It uses an adaptive Kalman filter to fuse the WiFi signal strength measurements from multiple access points with the inertial data, and also takes the map constraints into account. The work in [64] uses particle filters to localise the users in indoor environments, where inertial information from foot-mounted IMUs and the building model are combined to produce accurate location. The footSLAM system in [65] applies a similar technique in the simultaneous localisation and mapping (SLAM) setting, and uses the Rao-Blackwellised particle filters to build the map of the environment and localise the users simultaneously. The Zee system in [66] considers an automatic signal strength fingerprinting application, and uses particle filters to fuse information generated by different users to produce the signal strength database.

Uncertain data management: Inference and state estimation techniques have also been used in managing uncertain sensor data. For example, the work in [67] proposed an approach that integrates particle filters directly with database views to support declarative queries about the estimated states. Underneath the system keeps all the weighted particles generated by inference and rephrases the user queries in such a way that they could be executed on the particle sets directly. The system also supports techniques like fixed-lag

smoothing to improve the inference results. The approach presented by Tran et al. [68] on RFID object tracking also uses such sample-based techniques but their enhanced approach with particle factorisation, spatial indexing and belief compression is more scalable for scenarios that contain large numbers of objects. Different from the above two approaches which use on-line inference over sensor data, the Lahar system [69] infers the sensor readings in an off-line fashion. The advantage of this approach is that inference over archived data could produce more accurate results than the on-line approaches, where the later observations can be exploited to refine the estimation of earlier data.

Distributed inference in sensor networks: Another branch of work on inference problems in sensor networks is distributed inference [70, 71]. Rather than gathering and processing data in a central point, e.g. a mobile device or a remote server, those approaches aim to perform inference at multiple sensor nodes in the network. The work in [70] uses distributed particle filters to infer the current system state at distributed sensor nodes while minimising communication overhead. In this work, each node maintains a local particle set and may share a small portion of randomly selected particles with the neighboring nodes. When receiving such particles, the node checks its own cache and broadcasts a set of most informative particles (chosen by certain distance metrics) to its neighbors, and the information of the global state can be gradually acquired by all the nodes. The work in [71] forms the sensor nodes as a junction tree [42] and infers the unknown states by exchanging messages of local measurements as in the typical junction tree algorithm, where the communication overhead can be minimised by tuning the topology of the sensor nodes.

2.1.3 Discussion

The proposed inference-based accuracy estimation approach is closely related to the above discussed inference and state estimation techniques. Firstly, the inference-based accuracy estimation approach models the physical signal and the measurements from multiple sensor systems as augmented hidden Markov models or linear dynamical systems, which are essentially special cases of the general state space models. Secondly, the inference-based accuracy estimation approach uses the idea of inference and state estimation as one important step when estimating the accuracy of sensor measurements: it infers the monitored states first and then uses the estimated states as pseudo ground truth to evaluate the measurement accuracy. Finally, in our experimental evaluation, approximate inference techniques such as particle filters are extensively used in the positioning systems we have built.

However, the proposed inference-based accuracy estimation approach is fundamentally different from the existing work. The goal of our approach is not to infer the latent states,

but to estimate the accuracy of the sensor systems. Comparing with the work using Kriging or Gaussian Processes [52, 54, 55] mentioned above, our approach does not use the regression-based techniques to estimate the states, but includes states and sensor measurements in a probabilistic model. The model developed by our approach is different from existing ones, such as the linear dynamical systems or hidden Markov models, which have been extensively used in the map matching techniques [56, 57, 19, 58, 59]. Our model is able to incorporate multiple sequences of probabilistic sensor measurements, and exploit the prior state distributions at arbitrary timestamps. Although some existing work has studied models with multiple observation sequences [2] or missing data [72], to our knowledge there is little work on models consuming multiple probabilistic observations. Moreover, there is very limited research on applying inference techniques to the problem of estimating the accuracy of coexisting sensor systems.

2.2 Learning and Parameter Estimation

Our work is also connected to research in learning and parameter estimation, since the learning-based accuracy estimation approach proposed in this thesis employs learning techniques to re-estimate the model parameters. Concretely, rather than relying on the model parameters known a priori, the proposed learning-based approach first learns the parameters that are the most consistent with the observed probabilistic sensor measurements and available prior knowledge. The learned model parameters are then used to infer the latent states, which can improve accuracy estimation. In some cases (as we will show in Chapter 5 later), the learned parameters can be used to derive the accuracy of sensor systems directly. Therefore, the key step in the proposed learning-based accuracy estimation approach is to estimate the model parameters given the observed data, which is closely related to the learning and parameter estimation techniques reviewed in the rest of this section. In the following, we also first explain the related learning and parameter estimation techniques in general graphical models, then survey their applications in sensor networks, and finally show how our approach is related to these existing techniques.

2.2.1 Learning and Parameter Estimation in Graphical Models

Besides inference, learning is another vital problem in graphical models, and it typically involves two tasks: one is *finding the model structure*, and the other is *estimating the model*

parameters. Usually the model parameters are always assumed to be unknown, and depending on whether the model contains hidden variables, learning in probabilistic graphical models (PGMs) involves the following four scenarios [73]: a) the model structure is known and the variables are fully observable; b) the model structure is known and the variables are partially observable; c) the model structure is unknown and the variables are fully observable; and finally d) the model structure is unknown and the variables are partially observable. Scenario a) is trivial, since in that case there is no hidden variable, and the Maximum Likelihood (ML) estimate of the model parameters can be directly computed by various optimisation tools, such as gradient descent [74], Gauss-Newton method [75] and conjugate gradient [76], etc. Cases c) and d) involve learning the model structure, which is often referred to as the *model selection* problem. There is a solid body of work focusing on this problem (we refer the readers to the books [29] and [30] for more details), but in this thesis, we consider state space models (SSM), whose structure is assumed to be known in advance.

As discussed in the previous section, a state space model contains two sets of variables, the latent variables (states) and the evidence variables (or observations / measurements). The states follow a stochastic process, which is observed through the evidence variables. Given the structure of the state space models, we typically consider the following model parameters: a) the initial state distribution, b) the correlations between different states (the transition model), and c) the dependencies between the states and the evidence (the measurement / emission model). To learn those parameters, the existing learning algorithms can be broadly divided into exact and approximate approaches. Note that exact learning algorithms only work on simple models, such as hidden Markov model (HMMs) or linear dynamical systems (LDSs) with Gaussian variables, while for more general state space models (SSMs) there is no closed form solution and approximate learning algorithms are used instead.

Generally, the learning problem in state space models (SSMs) can be addressed by the Expectation Maximisation (EM) scheme, which iteratively performs the E-step and the M-step until converging to a local optimum. In each iteration, the current estimated model parameters are used in the E-step to derive the expected likelihood of the data (often referred to as the likelihood function), and the M-step finds the new model parameters that maximise the previously derived likelihood function. Typical convergence criteria for the general EM scheme are: a) the likelihood of the observed data does not increase significantly, b) the new parameters stop changing significantly, or c) a certain predefined step count is reached.

For hidden Markov models (HMMs), the most well-known exact learning algorithm is the Baum-Welch algorithm [77], which is a special case of the EM scheme. For linear dynamical systems (LDSs) with Gaussian variables, the Kalman learner which is based on the standard Kalman filter and smoother [38, 45] is commonly used. For models which are more complex than the simple HMMs and LDSs, a large number of approximate learning algorithms have been developed. In fact, as discussed above, the EM framework typically include inference (smoothing) as a subroutine in its E-step, while the M-step usually incorporates suitable optimisation techniques to find the new model parameters. Therefore, most of the existing algorithms address this problem by fitting approximate inference techniques into the EM framework. For example, the extended Kalman learners use the extended Kalman smoother to evaluate the likelihood of data in the E-step [78]. Other approximate inference approaches, such as the sample-based techniques or variational Bayes, can be incorporated in the EM scheme in a similarly way [46, 79, 80]. Another alternative of approximate learning is to combine Gaussian process (GP) with the EM scheme, as shown in [81].

2.2.2 Learning and Parameter Estimation in Sensor Networks

Learning and parameter estimation has attracted considerable interest in the sensor network community. While much of the initial work focused on applying parameter estimation techniques in the context of sensor calibration [82], more recently there have been many attempts to apply them in a broad range of problems, e.g. sensor node localisation [83], network topology learning [84], target tracking [85], environmental mapping [86], and social sensing [87].

Sensor calibration: Sensor calibration has been an active topic since the very beginning of sensor network research and a solid body of work has been proposed to address this problem. Unlike most of the work which treats sensors individually and calibrates each of them separately based on its output, the work in [82] casts the calibration problem into that of a parameter estimation problem. Concretely, this approach first models the input / output of the sensor network with a set of parameters (associated with each node), then it optimises the response of the entire network with respect to those parameters based on collected training data. The advantage is that well-established techniques in parameter estimation can be borrowed to provide a unified sensor calibration framework. The work in [88] extends this idea to the distributed setting, and calibrates sensors with a consensus-based parameter estimation algorithm. This algorithm finds the parameters that minimise

the expected differences in the output of neighbour nodes, so that the majority of well-behaved sensors will correct the drifted ones (if no external reference is given).

Sensor localisation and target tracking: Another popular class of applications where learning and parameter estimation techniques have been widely used includes sensor localisation and target tracking [83, 89, 85]. It is typically assumed that the sensor nodes are sparsely distributed, where one or more moving targets are passing the monitored area. The locations of the stationary nodes are used as references to determine the trajectories of the moving targets, while the observations of the targets can also provide useful information on the locations of the stationary nodes. Therefore, most of the existing work addresses the self-localisation of the sensor nodes and target tracking simultaneously. For instance, the work in [89] proposes a centralized algorithm, which models the self-localisation and target tracking problem as the parameter estimation problem in state space models. The locations of the sensors are considered as model parameters, while latent states are the locations of the targets. It uses a sequential Monte Carlo technique (the auxiliary particle filter) to infer the target locations, while simultaneously estimating the locations of the sensors with on-line Expectation Maximisation (EM) scheme.

The work in [83] addresses the same problem in a distributed way. It casts the sensor self-localisation problem into a parameter estimation problem for Hidden Markov models (HMMs), and proposes both a distributed recursive maximum likelihood (ML) algorithm and a distributed Expectation Maximisation (EM) algorithm to estimate the parameters. Those distributed algorithms rely on a message passing scheme which exchanges the belief between different nodes to achieve the consensus state. During the parameter estimation process, the trajectories of the tracked targets can be determined simultaneously. For non-linear cases, [83] uses a distributed learner based on the extended Kalman filter (EKF), which relies on the local linearisation of the model and approximates the model parameters accordingly.

Learning the network topology: The work in [84] considers the problem of learning the topology of the network in the context of camera networks, whose sensors are completely non-overlapping. It assumes that the camera network is tracking multiple agents, whose movements are modeled by semi Markov processes. The camera network is modeled as a directed graph, where the vertices indicate the locations of the sensor nodes and the edges represent the connectivity between them, i.e. the topology of the network that needs to be learned. This work includes such network topology, as well as the expected duration between state transitions, as the parameters of the semi hidden Markov model, and proposes an algorithm that combines the Markov chain Monte Carlo techniques and the expectation maximisation (EM) scheme to estimate those parameters. More recent work [86] also

considers the problem of simultaneous localisation and environmental mapping, which estimates the location of the sensors by looking at the values of the environmental variables monitored by the network. It assumes that the variables such as temperature, are spatially correlated (the correlations depend on the geographical distances), and change smoothly over time. It models the environmental variables as the correlated latent states, and the locations of the sensors as the model parameters. It then uses the EM scheme to iteratively find the model parameters, i.e. sensor locations, that can best explain the sensor measurements.

Truth discovery in social sensing: Recently, the techniques of learning and parameter estimation are also used in the context of social sensing and crowd sourcing [87, 90]. For example, the work in [87] considers a sensing scenario where a large number of users report a physical phenomenon, e.g. litter in a park. It tries to address the problem of estimating both the correctness of the measurements and the reliability of the participants. It fits this problem into the EM framework, and considers the correctness of measurements and the reliability of the participants as parameters of a Bayesian model. The reliability of a participant is represented by the likelihood of the participant reporting an event given the fact that it actually has happened, i.e. the probability of genuine report, while the correctness of observations is the posterior distribution that the event happens given that it has been reported. Then it iteratively finds the parameters that maximise the likelihood of the observed data until convergence. Note that this approach works off-line (in a batch fashion), in the sense that the EM algorithm needs to process the entire set of observed data when estimating the model parameters.

On the other hand, the work in [90] considers a stream setting in a similar social sensing scenario, where the observations reported by different users are generated continuously. It tries to address the problem of estimating the correctness of observations and the reliability of the participants on-line. Similar to [87], this approach also represents the two quantities as parameters in a Bayesian model. The main difference of [90] compared to [87] is that, rather than operating the Expectation Maximisation (EM) scheme in a batch mode, i.e. waiting until all data has been observed, this approach uses a recursive version of EM scheme: the parameters are updated continuously based on the previously evaluated parameters and the newly observed data. A vital assumption of this approach is that the parameters, particularly the participant reliability, only change slowly over time, otherwise it would be difficult for the EM scheme to converge. Another major difference is that when a new observation arrives, this approach only runs the EM scheme for one iteration rather than multiple times to avoid over-fitting the new data.

2.2.3 Discussion

The proposed learning-based accuracy estimation approach has many connections with the learning and parameter estimation approaches discussed above. Firstly, the model used by the proposed learning-based approach is an extension of standard state space models, e.g. HMMs or LDSs: particularly in how it represents the conditional dependencies between the monitored states and the sensor observations. Secondly, our learning-based approach uses the idea of parameter estimation to perform the task of accuracy estimation. In particular, our approach estimates the model parameters, and uses the learned parameters to either a) improve the state estimation and thus further improve the quality of accuracy estimation, or b) directly reason about the accuracy of sensor systems. Thirdly, the learning algorithm we use in the proposed approach follows the classic EM framework, which has been widely used in existing work. Finally, our approach falls into the class of maximum likelihood estimation (MLE) approaches to ascertaining sensor reliability, which bears close resemblance to the truth discovery techniques in social sensing [87, 90].

However, the proposed learning-based accuracy estimation approach is also quite different from the above surveyed existing work. Unlike existing work, which uses the parameter estimation techniques to perform sensor calibration or learn the static properties of sensor networks (e.g. node topology and locations, or sensor noise profiles) [82, 83, 84, 86], the goal of our learning-based approach is to evaluate the accuracy of multiple sensor systems. In that sense, the truth discovery approaches [87, 90] share similar objectives with the proposed learning approach, where both approaches try to estimate the reliability of the information sources (e.g. human participants in their social sensing scenarios, and sensor systems in our cases) based on the observed data. However, the proposed learning-based approach is different from the existing truth discovery approaches in many ways.

Firstly, the problem formulation of the two approaches is very different. The truth discovery approaches consider a simple Bayesian model which ignores the temporal correlations between the states, while our approach uses augmented state space models to exploit the underlying dynamics. Also, our approach assumes that the information sources (sensor systems) provide uncertain measurements (probability distributions), which is naturally more general than the truth discovery techniques, which only consider deterministic binary observation (i.e. true or false). Moreover, the proposed learning-based approach exploits prior knowledge on state distributions, which is not considered in the truth discovery approaches. Secondly, the truth discovery techniques assume that the source reliability changes slowly over time and space, while our approach does not make such limiting assumptions, and exploits the fact that the accuracy of a sensor system can vary greatly in different contexts. Thirdly, the truth discovery approaches typically tend to work well

when a large number of sources are used, e.g. in the social sensing scenario, whereas our approach can work well in scenarios where only a few coexisting sensor systems are available (e.g. less than five). Finally, although both approaches are built on the general Expectation Maximisation (EM) framework, the derivation of our algorithm is very different from the truth discovery approaches: they consider the standard EM algorithm on Bayesian models, while our approach extends existing EM algorithms for dynamic models with probabilistic observations.

2.3 Other Related Work

In this section, we overview other research efforts that are closely related to our problem of estimating the accuracy of sensing systems, but do not fall in the general classes of state inference or parameter learning.

Indoor positioning: This thesis uses indoor positioning as one of the running examples to explain and evaluate the proposed accuracy estimation approaches. Particularly, the proposed work is related to three categories of positioning systems: a) systems that use RSS (Received Signal Strength) -based approaches; b) systems that adopt motion sensing techniques; and c) systems that fuse both motion and RSS information to infer positions. Here we mainly review the existing work in classes a) and b), where systems belong to class c) has been already discussed in inference-based sensor tracking (Section 2.1.2).

RSS-based positioning systems have become popular solutions due to the wide availability of wireless access points (e.g. WiFi APs), and the cost benefits of not deploying extra infrastructure. There are mainly two classes of RSS-based positioning techniques [91]. One is based on radio propagation models, and the position of a target is determined by calculating the distances (with the propagation models) between the target and multiple APs with known locations, such as [92], [93] and the SNAP-WPS system [94]. The main disadvantage of this class of approaches is that radio propagation in indoor environments is very difficult to model due to the mixture of LOS (Line-of-sight) / NLOS (Non-line-of-sight) signals, and our recent work (e.g. [95]) has proposed a machine learning technique to identify and mitigate the NLOS conditions. The other class of approaches is fingerprinting, which does not rely on radio propagation models, but leverages the mapping between RSS measurements and coordinates obtained from a training phase. Early systems include Radar [96], PlaceLab [97] and Horus [98]. Recent techniques have been proposed to improve fingerprinting performance, e.g. by incorporating heterogeneous wireless

clients [99], or exploiting additional features of the environment, such as FM signals [100], sound, light and color [101]. However, one limitation of this class of methods is that they typically require labour-intensive training phases to generate the radio maps, while recently a number of techniques have been proposed to automatically build the radio map by fusing RSS measurements with motion sensor data, such as [66], [62] and [102].

Another category of positioning techniques that is closely related to this thesis is motion sensing, which calculates the current position of the tracked person based on odometry data. The data could be measured by either the Inertial Measuring Units (IMUs) mounted on the feet of a pedestrian [103, 64, 104], or the on-board inertial sensors of the mobile device carried by the person [105]. Most of the existing motion sensing techniques include the following three steps: 1) motion mode recognition, which detects different modes of movement (e.g. stationary vs. walking) [105]; 2) displacement estimation, which uses the acceleration data to estimate the distance traversed during a given period (e.g. step length estimation [106]); and finally 3) orientation tracking, which uses data from magnetometer, accelerometer and optionally, gyroscope to estimate the orientation of the device [107, 108]. A major challenge for motion sensing is that the error in the inertial measurements accumulates over time, and can also be exacerbated by many factors, e.g. different walking styles or magnetic field distortions. Recent work tries to tackle this challenge and improve the quality of motion sensing: e.g. [109] and [110] exploit encounter information between a group of tracked users to correct the drifted trajectories, while the work in [111] has proposed an inertial tracking technique using mobile phones, which is robust to the variability in position / orientation of the phones and walking profiles of different users.

Estimating the accuracy of sensor data: In most existing work, the accuracy of sensor data is evaluated with the knowledge of ground truth, e.g. the accuracy of a positioning system is typically defined as the Root-Mean-Square error (RMSE) of the estimated trajectories with respect to the actual trajectories ([112] has provided a thorough discussion on the accuracy of various existing positioning systems). One area of research which is closely related to this thesis, is to reason the accuracy of sensor data without possessing the ground truth. There has been a solid body of sensor network research along this line. For instance, early work [113] considers the scenarios where multiple sensors are monitoring the same signal, and proposes a statistical approach to detect the inaccurate measurements. However it does not explicitly quantify the accuracy of sensor data. The work in [114] assesses the accuracy of sensors with simple majority voting scheme, while [115] uses the sensor noise profiles and known characteristics on monitored signals to estimate the accuracy. On the other hand, the work in [116] considers social sensing applications where the

reliability of human participants are not available, and proposes both real and asymptotic accuracy bounds of the reported measurements. Recent work [117] proposes a model-based approach to estimate the accuracy of environmental sensors, which can detect systematic and transient sensor errors. The work in accuracy-energy aware localisation [118, 119] is also related to our work, which seeks the best trade-off between localisation accuracy and energy consumption when tracking a mobile device. However, it is different from our work since their accuracy models of the sensors are either derived from prior knowledge or provided by the underlying Android operating system.

Fact finding in information networks: The accuracy estimation approaches proposed in this thesis are also closely related to the quality estimation approaches, e.g. fact finding techniques in information networks [120, 121, 122]. In these networks, sources and assertions are represented as nodes, and each fact "source i made an assertion j" is represented by a link. Nodes are then assigned credibility scores in an iterative manner: for example, in a basic fact finder [120], an assertion's score is set to be proportional to the number of its sources, weighted by the sources' scores; similarly, a source's score is set to be proportional to the number of the assertions it made, weighed by the assertions' scores. A Bayesian interpretation of fact finding is offered in [123] that allows *quantifying the actual probability that a source is truthful or that an assertion is true*. Whereas we share the same goal of assessing the credibility of different data sources, we cannot directly apply fact finding techniques. The key reason is that fact finding techniques tend to work well when a large number of sources are used to report on the same state (e.g. social sensing), and is therefore not suitable for traditional sensor networks, where only very few sensors typically detect and report the same event.

Cost-constrained data acquisition in sensor networks: Another related area of research is cost-constrained data acquisition in sensor networks. Most work in this area explores the trade-off between the quality of information and the costs incurred in retrieving the sensor data, and it can be broadly divided into two branches. The first branch assumes that sensor readings concern the same variable of interest and are typically correlated over space. The goal is then to place sensors, or select sensors to be active, at informative locations, e.g. using Gaussian Process (GP) models of the sensed phenomenon [124, 125, 126, 127]. The second branch of cost-bounded data acquisition considers heterogeneous sensors, and applications where the variable of interest may not be directly measured by a sensor, but rather derived from a variety of sensor measurements using a prediction model. An example is the study of selecting suitable regression models to minimise the prediction error given a cost budget [128]. Our work is different from the above cost-constrained data acquisition techniques in two ways. Firstly, the sensor systems in our context are typically

black boxes, and we have little or no knowledge on where the sensors are placed, and thus cannot use this information to build the Gaussian process models. Secondly, in our work the ground truth is assumed to be absent, and therefore we cannot use knowledge on the real states to train the regression models.

Recommendation systems: The accuracy estimation approaches proposed in this thesis also share some common ideas with recommendation systems, which have been widely used in on-line retailer systems such as Amazon or eBay. A recommendation system leverages the information on the previous behaviour of the user (content-based filtering [129]), or the experiences from other users (collaborative filtering [130]), or a combination of them (hybrid filtering [131]), to infer the preference of the user and provide recommendations accordingly. The recommended items are listed with different weights / scores, typically in descending order of how relevant they are. The proposed accuracy estimation approach is similar in the sense that it considers the accuracy of the coexisting sensor systems as their scores, and recommends the more accurate system for the sensing application to task. However, our work is different because the accuracy of sensor systems is evaluated based on information provided by all available systems, where as recommendation systems typically compute scores based on relevance between the experiences of the user and the feature of items. In fact, our work can be potentially extended to incorporate the ideas of recommendation systems, where previous experiences on using the sensor systems may help the sensing applications to decide which system to use in the near future.

Chapter 3

The Accuracy Estimation Framework

This chapter proposes an *accuracy estimation framework*, which lies between the sensing application and the underlying sensor systems, and estimates the accuracy of the sensor systems given the requirements of the application. The proposed framework provides a general solution for accuracy estimation, and can be applied to a broad spectrum of sensing scenarios. The framework is also flexible in the sense that its layers can be easily modified or merged to adapt to various settings, and each of the layers can be implemented with different techniques. We begin with an overview of the proposed framework in Section 3.1, then Sections 3.2 $\sim$ 3.5 explain the individual layers of the framework in more detail, and in Section 3.6 we show how the framework can be implemented with the running example of an environmental monitoring scenario. In Chapters 4 and 5, we discuss the inference-based and learning-based accuracy estimation approaches, which are two different implementations of the proposed framework.

3.1 Overview

In the presence of multiple systems providing sensing services, the task of the accuracy estimation framework is to estimate their accuracy based on the accuracy requirements $\langle f_\epsilon, \boldsymbol{A} \rangle$ provided by the application, as discussed in Chapter 1. Intuitively, the accuracy of a sensor system should depend on the accuracy of its measurements, and the accuracy of a given measurement z_t^m is defined by the accuracy metric f_ϵ, with respect to the ground truth x_t. Therefore, the real accuracy of a sensor measurement z_t^m is the output of the function $f_\epsilon(z_t^m; x_t)$. In our context, since the state x_t is unknown, the real accuracy of a measurement $f_\epsilon(z_t^m; x_t)$ cannot be evaluated directly. However, if the state x_t in $f_\epsilon(z_t^m; x_t)$ can be fairly estimated, the accuracy of a sensor measurement can be approximated by

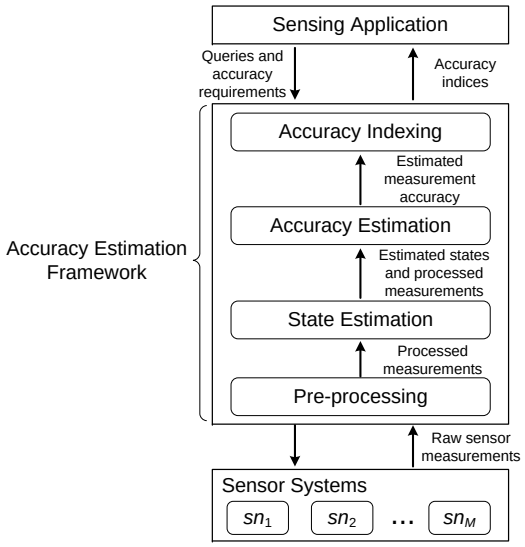


Figure 3.1: The architecture of the proposed accuracy estimation framework, where measurements reported by the sensor systems are passed through four layers: pre-processing, state estimation, accuracy estimation, and accuracy indexing. The output of the proposed framework is the accuracy indices of the sensor systems, as specified by the requirements $\langle f_\epsilon, \boldsymbol{A} \rangle$ of the sensing application.

feeding the estimated state $\hat{x}_t$ into the accuracy function f_ϵ. Following this intuition, the proposed accuracy estimation framework contains four layers:

- **Pre-processing layer:**- This layer takes the raw measurements generated by the sensor systems as input, and processes them so that they conform to the same clock, metric system and probabilistic form.

- **State estimation layer:**- This layer takes the output of the pre-processing layer as input, and computes the estimated states $\hat{x}_t$ given the pre-processed measurements from all sensor systems and the available prior knowledge on state distributions.

- **Accuracy estimation layer:**- This layer estimates the accuracy of the sensor measurements z_t^m, given the estimated states $\hat{x}_t$ (computed by the state estimation layer) and the accuracy function f_ϵ provided by the sensing application.

- **Accuracy indexing layer:**- This layer aggregates the accuracy of sensor measurements z_t^m (evaluated by the accuracy estimation layer) and builds the accuracy index for each sensor system sn_m according to the attributes $\boldsymbol{A}$ specified by the sensing application.

Figure 3.1 illustrates the layers of the proposed framework. As discussed in Chapter 1, the sensing application on top requires information on some physical signals, e.g. the temperature of a building or the locations of a user, and would like to select one sensor system to task. The sensing application poses corresponding queries to the proposed framework, and also specifies an accuracy metric and a set of attributes, which will be used to evaluate and index the accuracy of the reported measurements. The proposed framework firstly activates all the coexisting sensor systems that can provide measurements on the queried states for a short period of time. The raw sensor measurements pulled from the systems are synchronised and resampled through the *pre-processing layer*. Then the processed measurements from all available systems are forward to the *state estimation layer*, where the queried states are estimated based on the sensor observations and available prior knowledge on states. The next *accuracy estimation layer* takes the estimated states together with the pre-processed sensor measurements as input. In this layer, the accuracy of sensor measurements is evaluated with respect to the estimated states (computed by the previous *state estimation layer*), given the accuracy metric provided by the sensing application. The evaluated measurement accuracy is then forwarded to the *accuracy indexing layer*, where an accuracy index is built for each sensor system, according to the accuracy of its measurements and the attributes specified by the sensing application. Finally, the accuracy indices of the sensor systems are reported as the output of the accuracy estimation framework, which empowers the sensing application on top to choose the desired system to use in different context. The proposed framework will initiate this process periodically to keep the accuracy indices updated. Now we are in a position to describe the individual layers of the proposed framework in more detail.

3.2 The Pre-processing Layer

The input of the pre-processing layer is the raw sensor measurements collected from the coexisting sensor systems, and it outputs the processed measurements to the next state estimation layer. This layer is optional but usually necessary in practice. Raw measurements from different sensor systems are typically heterogeneous, for instance they may not conform to a global clock, or they may be generated at different time or space granularities. Moreover, some sensor systems may only be able to provide deterministic measurements without any error bounds. To address this, the pre-processing layer processes the raw sensor measurements through two steps: a) the *synchronisation* step, and b) the *resample* step.

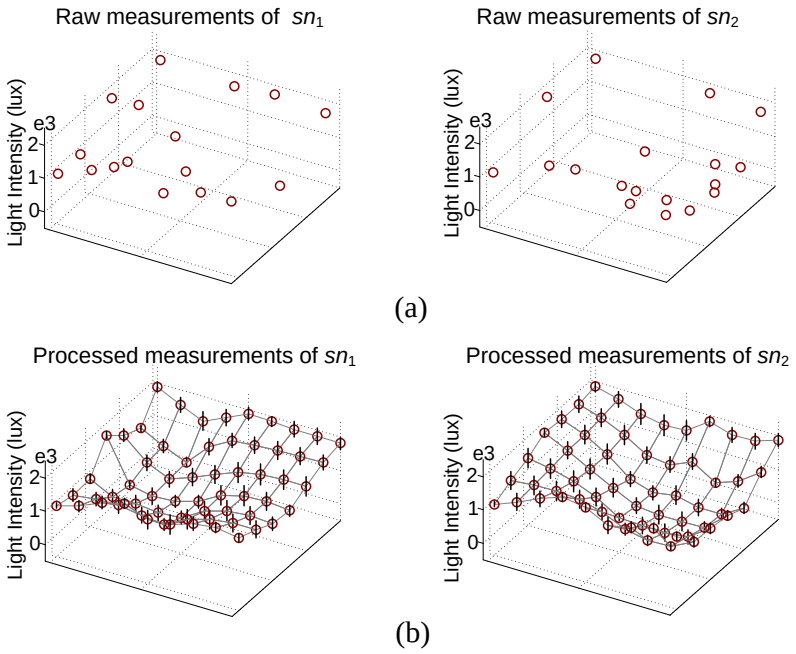


Figure 3.2: (a) A snapshot of the raw sensor measurements on light intensity generated from two sensor systems sn_1 (on the left) and sn_2 (on the right). (b) A snapshot of the pre-processed light intensity measurements from sn_1 (on the left) and sn_2 (on the right) by Gaussian process non-linear regression.

3.2.1 Synchronization

In the synchronization step, the measurements from different sensor systems are firstly re-timestamped according to the global clock provided by the proposed framework. If the measurements are generated in different metric systems, e.g. some temperature readings may be reported in Celsius scale while others could be in Fahrenheit scale, they are converted into the same metric system. For measurements without error bounds reported, this step also generates the corresponding error estimations and thus parses the deterministic measurements into probabilistic ones. This can be achieved by various existing techniques, such as model-driven approaches [117, 132], or approaches that leverage prior knowledge on sensor noise characteristics [115].

3.2.2 Resample

After the synchronization step, the resample step further subsamples or interpolates the measurements so that measurements from different sensor systems have the same time and space granularity. There is also a solid body of techniques that can be used in this context, and in our experiments, we use Gaussian process non-linear regression [53] to interpolate the sensor data over time and space. Figure 3.2(a) shows the raw sensor measurements on

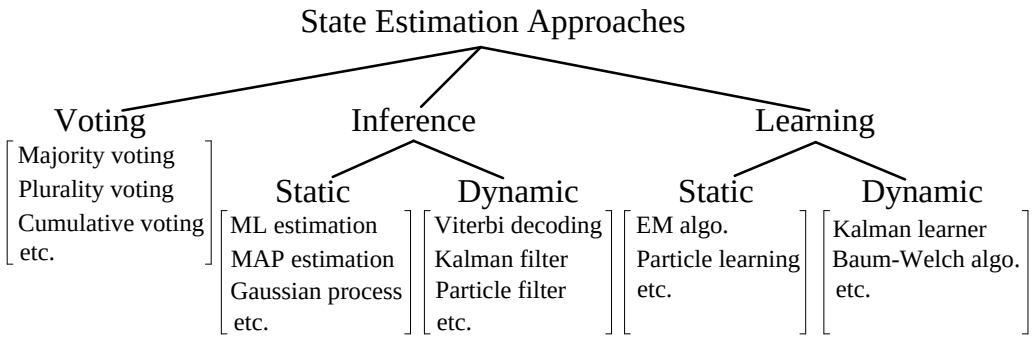


Figure 3.3: The taxonomy of state estimation approaches for sensor networks. For each line of approaches, several representative techniques are listed.

light intensity, generated by two coexisting sensor systems used in our experiments. Note that each system only generates deterministic measurements at the locations where they have sensors. Figure 3.2(b) shows the pre-processed sensor measurements, which have the same space and time granularity, and are annotated with confidence intervals.

3.3 The State Estimation Layer

Given the pre-processed sensor measurements and the prior state distributions as input, the state estimation layer estimates the actual states x_t, which are forwarded to the next accuracy estimation layer together with the sensor measurements. Formally, we assume that after pre-processing, at any given timestamp t, each sensor system sn_m, $1 \leq m \leq M$, provides a probabilistic measurement z_t^m of the state x_t, where z_t^m is a random variable with probability distribution $p(z_t^m)$. We also assume that certain prior distribution ρ_t on the state may be available at some of the timestamps, as defined in Section 1.4.1.4.

The state estimation layer can be implemented in various ways, and we design a taxonomy consisting of three main classes of approaches: voting, state inference and learning. The inference-based and learning-based approaches can be further divided into two subclasses: static and dynamic, depending on whether the dynamics of the monitored process are taken into consideration. Figure 3.3 illustrates the taxonomy, along with specific examples of techniques under each class. In the following text, we provide an overview of the taxonomy, and describe the high level ideas for each of the approaches. The detailed derivations of the inference-based and learning-based approaches will be explained in Chapters 4 and 5 respectively.

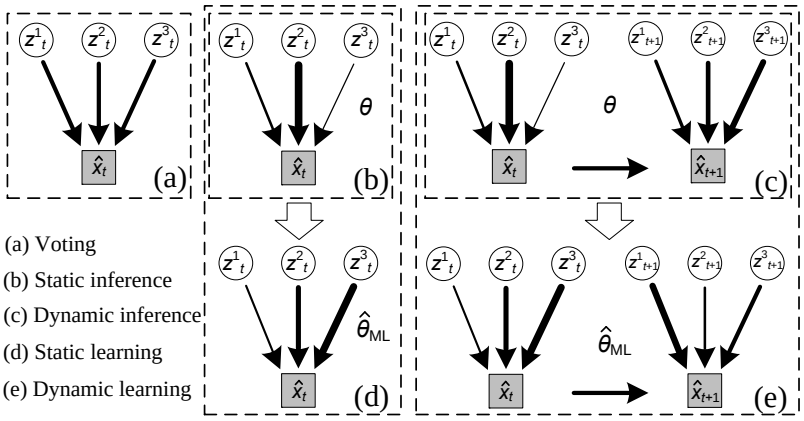


Figure 3.4: Comparison of different state estimation approaches. (a) Voting merges sensor measurements at t with equal weight to estimate $\hat{x}_t$. (b) Static inference infers $\hat{x}_t$ with measurements at t and the known model parameters θ, which determine the weights of different measurements (indicated by the weighted arrows in the figure). (c) Dynamic inference estimates $\hat{x}_t$ with the full sequence of measurements and model parameters θ, which also include the transitions between states. (d) Static learning re-estimates the model parameters with the measurements at t, and uses the learned $\hat{\theta}_{\mathrm{ML}}$ (indicated by the weighted arrows at the bottom, which are different from those in (b)) to infer $\hat{x}_t$. (e) Dynamic learning learns the model parameters with the full sequence of sensor measurements, and estimates $\hat{x}_t$ with the new parameter $\hat{\theta}_{\mathrm{ML}}$.

3.3.1 Voting-based approach

Voting is a widely used approach, which aggregates the preferences from multiple information sources to achieve a collective decision. In the context of state estimation, the voting approach works on one timestamp at a time. It treats the sensor systems as individual *voters*, and a probabilistic measurement z_t^m is the *voting plan* of the system sn_m, which distributes a fixed amount of scores (the probability mass) across the sample space Ω according to the measured distribution $p(z_t^m)$. The probability distribution of the estimated state $\hat{x}_t$ is then evaluated by combining the measured probability distributions $p(z_t^m)$ from the M coexisting sensor systems (using equal weights) according to certain voting rules, such as cumulative voting [133]. Figure 3.4(a) shows the basic idea of the voting-based approach.

3.3.2 Inference-based approach

The inference-based approach models the monitored states, the measurements from one or more sensor systems, and the prior state distributions as probabilistic models. The parameters of the model are assumed to be known a priori, and the latent states are estimated based on the observed measurements and prior distributions on the states. Depending on whether

we exploit the temporal correlations between states, the approach can be further divided into *static* and *dynamic* inference.

3.3.2.1 Static inference

Static inference ignores any temporal correlations between the latent states, and only accesses the measurements and prior distributions at single timestamps. Given the sensor measurements $z_t^{1:M}$ and prior ρ_t observed at t, it computes $\hat{x}_t$ as the posterior state estimate given the observed data, i.e. $p(\hat{x}_t) = p(x_t|z_t^{1:M}, \rho_t)$. If we assume the measurements from different sensor systems are conditionally independent and ignore any external noises, the distribution $p(\hat{x}_t)$ can be computed as:

$$p(\hat{x}_t) \propto p(\rho_t, x_t) \prod_{m=1}^{M} p(z_t^m|x_t) \tag{3.1}$$

where $p(z_t^m|x_t)$ $(1 \leq m \leq M)$ is the probability of observing the probabilistic measurement z_t^m given x_t. We assume that $p(z_t^m|x_t)$ is determined by the observed probability distribution $p(z_t^m)$ and the model parameters θ, which are assumed to be known a priori in this case. Note that the form of θ depends on the structure of the model, and Chapter 4 will describe the parameters θ for different types of models we use in more detail. For now we just assume that θ is a set of parameters which controls the probability distribution $p(z_t^m|x_t)$ given the measurement z_t^m. Conceptually, static inference involves the weighted fusion of the sensor measurements, where the weights are specified by the model parameters, as shown in Figure 3.4(b). If there are prior state distributions available at time t, $p(\rho_t, x_t)$ can carry such knowledge and bias the estimated state.

3.3.2.2 Dynamic inference

Dynamic inference assumes the monitored states are temporally correlated, and estimates the states with all the observed measurements and priors. Let the states be a time-varying sequence: $x_{1:T}$, with M measurement sequences $z_{1:T}^1, \ldots, z_{1:T}^M$ from the sensor systems. In practice, the dynamics are usually assumed to be *Markovian* for simplicity, i.e. $p(x_t|x_{1:t-1}) = p(x_t|x_{t-1})$, and the measurements at different timestamps are considered to be independent conditioned on the states. Under those assumptions, the distribution of the estimated state $\hat{x}_t$, i.e. the posterior state distribution given all the observed sensor measurements and priors, can be represented as:

$$p(\hat{x}_t) \propto p(x_t|z_{1:t}^{1:M}, \rho_{1:t}) p(z_{t+1:T}^{1:M}, \rho_{t+1:T}|x_t) \tag{3.2}$$

where $p(x_t|z_{1:t}^{1:M}, \rho_{1:t})$ and $p(z_{t+1,T}^{1:M}, \rho_{t+1,T}|x_t)$ can be evaluated iteratively:

$$\begin{aligned} p(x_t|z_{1:t}^{1:M}, \rho_{1:t}) \propto & \\ & \int_{x_{t-1}} p(z_t^{1:M}|x_t)p(\rho_t, x_t|x_{t-1})p(x_{t-1}|z_{1:t-1}^{1:M}, \rho_{1:t-1})\,\mathrm{d}x_{t-1}; \\ p(z_{t+1:T}^{1:M}, \rho_{t+1:T}|x_t) \propto & \\ & \int_{x_{t+1}} p(x_{t+1}, z_{t+1}^{1:M}, \rho_{t+1}|x_t)p(z_{t+2:T}^{1:M}, \rho_{t+2:T}|x_{t+1})\,\mathrm{d}x_{t+1} \end{aligned} \tag{3.3}$$

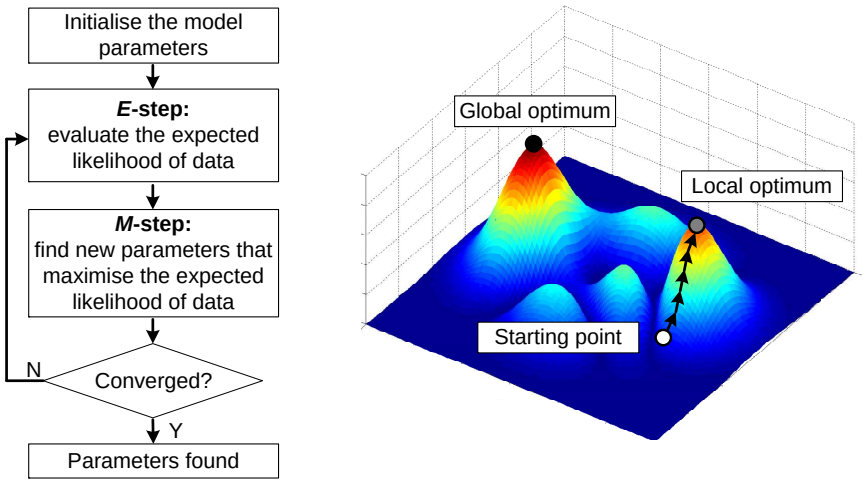
where $p(z_t^{1:M}|x_t)$ can be factored as shown in the static case, and is determined by the observed probability distributions $p(z_t^1)$, ..., $p(z_t^M)$ and the model parameters θ. Typically, the probability distributions $p(x_{t+1}|x_t)$ that links states at two consecutive timestamps are also considered as parts of the model parameters θ. Therefore in the dynamic case, the model parameters θ determine both the transitions between states, and the likelihood of observing sensor measurements given the state. Note that the priors are incorporated in the term $p(\rho_t, x_t|x_{t-1})$, which biases the state transition governed by model parameters θ towards the priors ρ_t. Figure 3.4(c) shows a simple case with two timestamps, where $\hat{x}_t$, $\hat{x}_{t+1}$ are estimated with measurements from three sensor systems and the model parameters θ.

3.3.3 Learning-based approach

Unlike the inference-based approach, the learning-based approach does not assume any prior knowledge on the model parameters θ. In the static case, the parameters θ contol the likelihood of observing the probabilistic measurements given the state, while in the dynamic case, θ also govern the correlations between adjacent states. The learning-based approach starts with an estimate of the parameters θ, and iteratively refines this estimate to be more consistent with the sensor measurements and prior state distributions. The estimated states $\hat{x}_t$ are then inferred with the learned model parameters and the observed data.

3.3.3.1 Static learning

Static learning first tries to find the model parameters θ that are the most consistent with the data observed within each timestamp, given by the maximum likelihood (ML) estimate $\hat{\theta}_{\mathrm{ML}}$, as shown in Figure 3.4(d). As in general latent variable models, $\hat{\theta}_{\mathrm{ML}}$ can be computed by the Expectation Maximisation (EM) approach [134]. The basic idea of the EM approach is that we firstly "guess" the unknown model parameters θ, and use this set of θ to evaluate the likelihood of the observed data. Then we try to see if there is another set of parameters



(a) (b)

Figure 3.5: (a) The workflow of the Expectation Maximisation (EM) scheme. (b) An example showing that the EM scheme may converge to a local optimum.

θ' that can better explain the observed data, i.e. increase the likelihood of data, and perform this for multiple times until we find the best set of parameters $\hat{\theta}_{\text{ML}}$. Therefore in practice, the EM scheme works iteratively with the following two steps until convergence:

1. The *E*-step, which computes the expected log likelihood function $Q(\theta', \theta)$ of the new parameters θ'. The expectation is taken with respect to the conditional distribution of the observed data given the states, under the current parameters θ:

$$\begin{aligned} Q(\theta', \theta) &= E_{x_t|z_t^{1:M},\rho_t,\theta}[\log p(z_t^{1:M}, \rho_t, x_t|\theta')] \\ &= \int_{x_t} p(x_t|z_t^{1:M}, \rho_t, \theta) \log p(z_t^{1:M}, \rho_t, x_t|\theta') \, \mathrm{d}x_t \end{aligned} \tag{3.4}$$

2. The *M*-step, which finds the new parameters θ' that maximise the Q function: $\theta' = \arg\max_{\theta'} Q(\theta', \theta)$. It can be proved that by the parameters θ' that maximise the Q function will always increase the likelihood of data [135].

Figure 3.5(a) shows the idea of the EM scheme. Note that the EM scheme only converges to a local optimum, and there is no guarantee that the global optimum can be reached. Figure 3.5(b) illustrates this situation, where from a starting point on the surface of data likelihood ($p(z_t^{1:M}, \rho_t, x_t|\theta')$ in this case), the EM scheme only converges to a local optimum after a number of iterations. Since this surface can be very complex in practice, we typically assume the converged parameters are the ML estimate $\hat{\theta}_{\text{ML}}$. The static learning approach then evaluates the distribution of the estimated state $\hat{x}_t$ with the learned model parameters $\hat{\theta}_{\text{ML}}$ as in static inference.

3.3.3.2 Dynamic learning

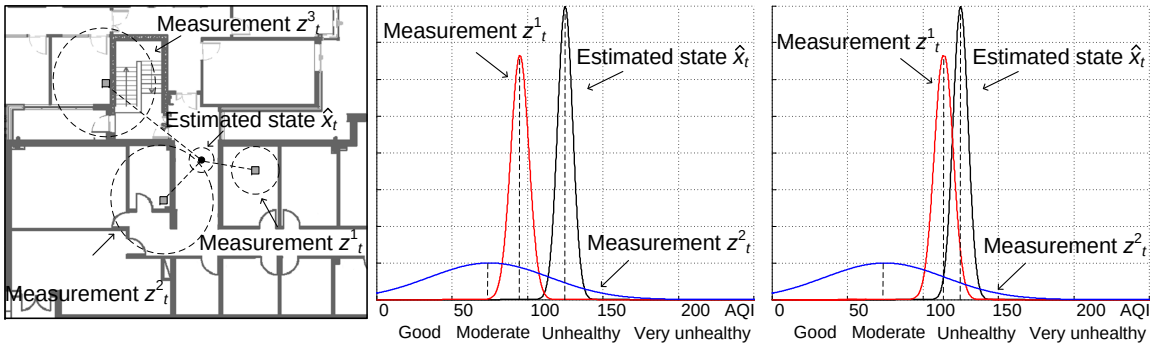
Similar to dynamic inference, dynamic learning also assumes the hidden state varies over time. But instead of relying on the model parameters known in advance, it firstly takes all measurements and prior state distributions into account and learns the parameters θ accordingly, as shown in Figure 3.4(e). Note that in the dynamic case, the model parameters θ that need to be learned include two parts, one that controls the state transitions, and one that governs the likelihood of observing sensor measurements given states. Dynamic learning also uses the EM scheme, but the derivation of the expected likelihood function Q is different from the static case. We make identical assumptions as in dynamic inference, and the likelihood function $Q(\theta', \theta)$ becomes:

$$\begin{aligned} Q(\theta', \theta) &= E_{x_{1:T}|z_{1:T}^{1:M}, \rho_{1:T}, \theta}[\log p(z_{1:T}^{1:M}, \rho_{1:T}, x_{1:T}|\theta')] \\ &= \int_{x_{1:T}} p(x_{1:T}|z_{1:T}^{1:M}, \rho_{1:T}|\theta) \log p(z_{1:T}^{1:M}, \rho_{1:T}, x_{1:T}|\theta') \, \mathrm{d}x_{1:T} \end{aligned} \tag{3.5}$$

Note that here we integrate over all possible state sequences $x_{1:T}$. The maximisation step is the same as in the static case. In both static and dynamic learning, the priors ρ_t are incorporated in each learning iteration when the Q function is evaluated, and thus bias both the estimated parameters and states.

3.4 The Accuracy Estimation Layer

The state estimation layer discussed above approximates the latent states with all observed sensor observations and the prior knowledge on state distributions. The estimated states and the processed sensor measurements are then forwarded to this accuracy estimation layer, where the accuracy of the sensor measurements is evaluated with the metric specified by the sensing application. As defined in Chapter 1, given a measurement z_t^m and the estimated state $\hat{x}_t$, the estimated accuracy of z_t^m is given by $f_\epsilon(z_t^m; \hat{x}_t)$, which is assumed to be a scalar value. The choice of f_ϵ plays an important role in accuracy estimation, and in this thesis we propose two accuracy metrics, a *proximity-based* and a *similarity-based* accuracy metric, which can be applied in sensing applications with different types of accuracy requirements. We introduce the two accuracy metrics in Sections 3.4.1 and 3.4.2, and compare them in Section 3.4.3.



(a) (b) (c)

Figure 3.6: (a) An indoor positioning scenario where three position measurements are reported. Under the proximity-based accuracy metric f^p_ϵ the measurement z^1_t is the most accurate, while z^2_t is more accurate than z^3_t. (b) An air pollution monitoring scenario where the air pollution level is determined by the AQI measurements. Under the proximity-based accuracy metric f^p_ϵ measurement z^1_t is more accurate, while under the similarity-based accuracy metric f^s_ϵ measurement z^2_t is more accurate. (c) Under both proximity-based and similarity-based accuracy metrics, measurement z^1_t is more accurate than z^2_t.

3.4.1 The proximity-based accuracy metric

The proximity-based accuracy metric defines the accuracy of a measurement based on its *distance* from the estimated state. For instance, one can simply use the Euclidean distance between the mean of reported sensor measurement and the mean of estimated state as the measure of accuracy. This thesis considers a more general metric, where the accuracy of a sensor measurement depends on its *expected Euclidean distance* from the estimated state. For a measurement z^m_t, its estimated proximity-based accuracy is defined as: $f^p_\epsilon(z^m_t; \hat{x}_t) = E[C(\|z^m_t - \hat{x}_t\|)]$, given the estimated state $\hat{x}_t$ and a cost function $C(\cdot)$. The cost function $C(\cdot)$ can take various forms, such as the quadratic cost function $\|z^m_t - \hat{x}_t\|^2$, and in this thesis, we consider the simplest form $\|z^m_t - \hat{x}_t\|$, where the estimated accuracy of z^m_t is given by:

$$\begin{aligned} f^p_\epsilon(z^m_t; \hat{x}_t) &= E[\|z^m_t - \hat{x}_t\|] \\ &= \int_{z^m_t, \hat{x}_t} p(z^m_t, \hat{x}_t) \|z^m_t - \hat{x}_t\| \, \mathrm{d}z^m_t \, \mathrm{d}\hat{x}_t \end{aligned} \tag{3.6}$$

From the definition, this accuracy metric $f^p_\epsilon(z^m_t; \hat{x}_t)$ tends to favour the sensor measurements whose probability mass is geometrically close to the ground truth, and is normally used in sensing scenarios that prefer such proximity. Figure 3.6(a) shows an example in the context of positioning systems, where given the estimated state $\hat{x}_t$, the proximity-based accuracy of sensor measurements z^1_t is higher than that of z^2_t since it has smaller variance (note that the means of z^1_t and z^2_t have the same distance to the mean of the estimated state

$\hat{x}_t$). On the other hand, although the measurements z_t^2 and z_t^3 have the same variance, z_t^2 is more accurate than z_t^3 since the probability mass of z_t^2 is closer to $\hat{x}_t$.

3.4.2 The similarity-based accuracy metric

The similarity-based metric defines accuracy as the *divergence* between the sensor measurements and the estimated states. There are many statistical divergence metrics that can be used in this context, and in this thesis we consider the widely adopted Kullback-Leibler (KL) divergence, which is a special case of the f-divergences. For a probabilistic measurement z_t^m, its estimated similarity-based accuracy is defined as the KL divergence between the distributions of the measurement and the estimated state:

$$\begin{aligned} f_\epsilon^s(z_t^m; \hat{x}_t) &= D_{\mathrm{KL}}(p(\hat{x}_t) \,\|\, p(z_t^m)) \\ &= \int_{u \in \Omega} \ln \frac{p_{\hat{x}_t}(u)}{p_{z_t^m}(u)} p_{\hat{x}_t}(u) \,\mathrm{d}u \end{aligned} \tag{3.7}$$

where $p_{\hat{x}_t}(u)$ is the probability density (probability mass in discrete cases) of distribution $p(\hat{x}_t)$ at $u \in \Omega$, and $p_{z_t^m}(u)$ is the density of $p(z_t^m)$ at u. f_ϵ^s indicates how well the measurement distribution $p(z_t^m)$ can be used to approximate the estimated state distribution $p(\hat{x}_t)$. This accuracy metric is particularly useful when the sensing application would like to discourage measurements with unreasonably high confidences. Figure 3.6(b) shows an example in an air pollution monitoring scenario, where the air pollution level is determined by the air quality index (AQI) measurements. Under the similarity-based accuracy metric, the measurement z_t^1 which is over-confident is less accurate than the measurement z_t^2, since most of the probability mass of z_t^1 lies in the category of "moderate" while the estimated state is in the "unhealthy" category.

3.4.3 Discussion on accuracy metrics

Of course many other functions can be used as accuracy metrics, and there is no universal accuracy metric that is suitable for all sensing scenarios. In practice, different sensing applications are likely to prefer different accuracy metrics, even if they are monitoring the same physical signal. For example, consider the air pollution monitoring scenario shown in Figure 3.6(b), where the sensor systems are monitoring the air pollution levels. If a sensing application would like to rank the measurements based on how close they are from the state, e.g. when calibrating the sensors, the proximity-based accuracy metric should be used: in Figure 3.6(b), z_t^1 is more accurate than z_t^2 under the proximity-based accuracy metric.

On the other hand, if a sensing application would like to classify the pollution level based on the reported measurements, the proximity-based accuracy metric may lead to wrong conclusions. In Figure 3.6(b), the proximity-based accuracy metric would prefer measurement z_t^1 since it is geometrically closer to the estimated state. However, it is undesirable for this application because as mentioned previously, z_t^1 puts the majority of its belief in the "moderate" (AQI 50-100) category, whereas the estimated state is actually "unhealthy" (AQI 100-150). This may cause serious false positives in practice, and the similarity-based accuracy metric can avoid this by favoring the measurement z_t^2 that reports a relatively "flat" distribution without concentrating its belief wrongly.

To generalize, the proximity-based accuracy metric cares more about the *distances* between the measurements and the estimated states, weighted by their probability distributions. Intuitively it would always reward a measurement whose mean value is closer to the estimated state and with a smaller variance, e.g. measurement z_t^1 in Figure 3.6(a). On the other hand, the similarity-based accuracy metric is more sensitive to the *differences* in the distributions, and is more suitable for applications that need to make certain decisions based on the distributions of the measured variables. It would prefer the measurements whose probability distributions (the way of distributing their probability mass) are more similar with those of the estimated states, i.e. share more mutual information. However, the two metrics are not necessarily always opposite. Figure 3.6(c) shows a similar scenario as in Figure 3.6(b), but in this case the measurement z_t^1 is considered to be more accurate under both proximity-based and similarity-based accuracy metrics.

3.5 The Accuracy Indexing Layer

The above accuracy estimation layer evaluates the accuracy of the individual sensor measurements, and this accuracy indexing layer further aggregates the estimated accuracy to reason about the accuracy of sensor systems, given the requirements from the sensing application. Intuitively, the accuracy of a sensor system should depend on the accuracy of all the measurements it generates. In practice, however, it is usually more meaningful to investigate the accuracy of a sensor system in a given *context*, e.g. accuracy during a certain period of time or at a particular location. In this thesis, we assume that the context information is specified by value assignments of a finite set of *attributes*. For instance, *time* is an important attribute for various types of sensor measurements; a position measurement from a localisation system is normally associated with a particular *user*, while a temperature reading is typically referred to a certain *location*. The proposed accuracy indexing

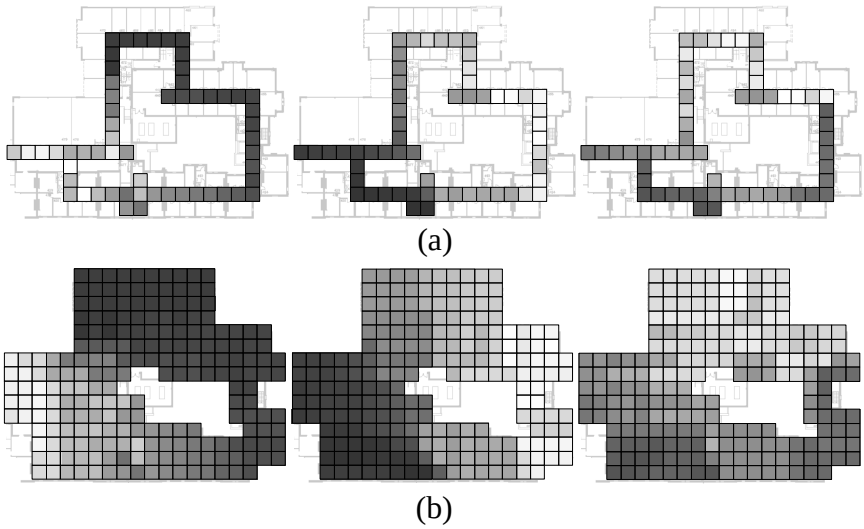


Figure 3.7: The accuracy indices over locations built for three sensor systems in the indoor positioning scenario, under the proximity-based accuracy metrics f_ϵ^p. The lighter blocks indicate smaller averaged f_ϵ^p values, i.e. the systems are more accurate at those locations. (a) After the accuracy aggregation step, the accuracy indices of the sensor systems only contain the averaged accuracy at the parts that have been visited by the user. (b) The accuracy indices after the accuracy interpolation step.

layer builds the accuracy indices in two steps: a) the *accuracy aggregation* step, and b) the *accuracy interpolation* step.

3.5.1 Accuracy aggregation

The accuracy aggregation step groups the sensor measurements by attributes $\boldsymbol{A}$ provided by the sensing application, and for each non-empty group it computes the average accuracy of all measurements in this group.

Let A be an attribute, such as time, user id, or location, etc. In this thesis we assume that the attributes associated with sensor measurements are discrete and with finite value domains. Let $a \in dom_A$ be a value defined in the domain of attribute A. We define the event that attribute A is assigned to value a as an *attribute assignment*, denoted as $A = a$, where $A = null$ indicates that there is no information known about this attribute. In practice, a sensor measurement is often associated with multiple attribute assignments, e.g. a temperature reading is typically annotated with both the timestamp and the location where it is observed. Therefore for a set of attributes $\boldsymbol{A}$, the event that $\boldsymbol{A}$ is assigned to a value tuple $\boldsymbol{a} \in dom_{\boldsymbol{A}}$ is defined as the *conjunction* of the assignments of each attribute $A \in \boldsymbol{A}$, denoted as $\boldsymbol{A} = \boldsymbol{a}$.

For a given sensor system sn_m, its accuracy under the attribute assignment $A = a$ is

the *expected accuracy* of the measurements that satisfy the assignment $A=a$. We denote this accuracy as $I_\epsilon^m(A=a)$. Conceptually, this accuracy $I_\epsilon^m(A=a)$ describes the expected behaviour of the sensor system given the specified context, e.g. the expected accuracy of the temperature measurements at a room, while in practice, $I_\epsilon^m(A=a)$ is typically evaluated with the set of measurements observed during the time period $1:T$. For sensor system sn_m, we refer to the subset of measurements whose attribute A is assigned to value a as the measurements *grouped by* the assignment $A=a$, denoted as $\boldsymbol{z}_{A=a}^m$. For example, it could be all the temperature measurements observed for a particular *location*, or during a certain *period of time*. Then given the measurements observed until time T, the accuracy of the system sn_m under $A=a$ is approximated as:

$$I_\epsilon^m(A=a) = g(f_\epsilon(z_t^m; x_t)), z_t^m \in \boldsymbol{z}_{A=a}^m \tag{3.8}$$

where g is the mean function that averages the accuracy of the sensor measurement and in this thesis we use the arithmetic mean. Then for all values $a \in dom_A$, the collection of accuracy values under all possible assignments forms the *accuracy index*, which in this case is an array of length $|dom_A|$. Each element in the accuracy index is $I_\epsilon^m(A=a)$. In the case where multiple attributes are considered, the accuracy index becomes a multidimensional array, or equivalently a cube, where each element is the accuracy of the sensor system sn_m under assignment $\boldsymbol{A}=\boldsymbol{a}$.

As a special case, if the value domain of the monitored states x_t is discrete and finite, it can be used as a special attribute to index the system accuracy as well. This is particularly useful when we want to profile the accuracy of a sensor system according to the *ground truth* of its measurements. For example, consider an indoor positioning system that reports the discrete locations of a user, i.e. the rooms or corridor segments that she is in. The performance of this positioning system can vary significantly over space, and it is crucial for the users to understand its positioning accuracy when they are moving across the indoor environment. In this scenario, the goal is to assess the accuracy of this positioning system in different locations, i.e. to profile the accuracy of the positioning system according to the real trajectory of the user.

However, unlike the other attributes, the ground truth of the measurements can not be determined exactly when the measurements are made. Therefore, in practice it is not possible to build the accuracy index by averaging the accuracy of measurements that shares the same ground truth. This thesis proposes a novel method that solves the joint problem of accuracy estimation and indexing with learning techniques, which is presented in Chapter 5. Figure 3.7(a) shows an example, where the sensing application would like to build

the accuracy indices of three positioning systems over the state space, i.e. the set of all possible discrete locations. Each block in Figure 3.7(a) represents the average accuracy of the observed measurements grouped by the particular location, where lighter blocks indicate higher accuracy.

3.5.2 Accuracy interpolation

However in some cases, the average accuracy computed by the aggregation step may not be sufficient to build the *complete* accuracy indices. For instance in the example shown in Figure 3.7(a), since the accuracy estimation process only runs for a limited period of time, the computed accuracy information may not be able to cover the entire environment, but only the parts that have been visited by the user. The accuracy interpolation step addresses this by filling up the unknown accuracy based on computed accuracy information. This step can be implemented by various techniques, such as linear/spline interpolation [136], wavelets [137], or Gaussian processes [53]. Figure 3.7(b) shows the complete accuracy indices after the interpolation step, where each location is annotated with the estimated accuracy.

3.6 Implementation Example

In this section, we discuss an illustrative example to show how the proposed accuracy estimation framework can be implemented in practice. Let us consider the environmental monitoring scenario introduced in Section 1.3.2 (this example will also be used to evaluate the proposed accuracy estimation approaches, and the detailed experimental setup will be discussed in Chapter 6). We assume that a sensing application is interested in certain environmental variables of an indoor environment, and here we only consider the light intensity for simplicity. The goal of the sensing application is to collect the most accurate light intensity measurements at different locations, e.g. in each room, under the similarity-based accuracy metric. On the other hand, two sensor systems sn_1 and sn_2, whose nodes are deployed in different parts of the environment (as shown in Figure 3.8), are able to provide measurements on light intensity. For simplicity we assume that the two systems are synchronised to a global clock, and the measurements are in the same metric system (e.g. lux). In this scenario, we assume the sensor measurements are *deterministic*, i.e. at a given timestamp a light intensity measurement reported is a scalar value, e.g. 500 lux. We also assume that some prior knowledge on the light intensity is available, for instance we may

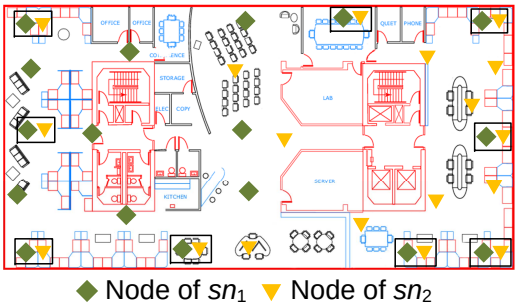


Figure 3.8: Node locations of the sensor system sn_1 and sn_2. sn_1 has more nodes on the left part of the environment, while sn_2 dominates the right part.

know that the server rooms are typically very dark (say 10 $\sim$ 200 lux), and the seminar rooms will be very bright (around 1500 lux) when meetings are in progress (according to the schedules of the rooms).

In this example, the sensing application poses queries on the light intensity to the accuracy estimation framework. It also specifies the similarity-based accuracy metric to use, and the attribute location (rooms) for accuracy indexing. In response, the proposed framework pulls data from both the two sensor systems sn_1 and sn_2, whose accuracy is evaluated and indexed as follows.

Pre-processing layer: In this particular example, the pre-processing layer is necessary, because a) the raw measurements from the sensor systems have different space granularities (each sensor system sn_1 and sn_2 only generates measurements at locations where it has sensors deployed as shown in Figure 3.8); and b) the reported raw measurements are deterministic and without any error bounds. Therefore, the goal of pre-processing layer is to a) resample the measurements into the same space granularity; and b) convert the deterministic measurements into probabilistic ones, which are annotated with confidence intervals. In our experiments, we use Gaussian process non-linear regression [53] techniques to pre-process the raw sensor measurements, so that at any given location both the systems sn_1 and sn_2 are reporting probabilistic measurements on local light intensity (the implementation details will be discussed in Section 6.1.2).

In practice, the implementation of this layer may vary, depending on different scenarios. From this example, we can see that the pre-processing layer can be omitted only if the measurements from all the coexisting systems are time synchronised, have the same space and time granularity, and conform to the same metric system and probabilistic form. It is possible for some sensing scenarios to not having this pre-processing layer, but typically this layer is needed when implementing the accuracy estimation framework.

State estimation layer: In this example, the state estimation layer takes the processed sensor measurements from systems sn_1 and sn_2 and the available prior knowledge as input, and estimates the actual light intensity at different locations. As discussed in Section 3.3, this state estimation layer can be implemented in various ways, from simple voting scheme, to more sophisticated inference-based and learning-based approaches. Suppose that in this example the state estimation layer is implemented with dynamic inference. In this case, the actual light intensity of a given location is considered as a stochastic process which evolves over time, and at each timestamp two probabilistic observations are reported from systems sn_1 and sn_2 respectively. Then the estimated states given all sensor measurements and priors, i.e. the estimated light intensity in this case, can be computed by Equation (3.2) and Equation (3.3) in Section 3.3.2.2. It is also worth pointing out that for different sensing scenarios, given that the reported sensor measurements are pre-processed properly, the implementation of this state estimation layer can be quite similar.

Accuracy estimation layer: The accuracy estimation layer takes both the estimated states, i.e. the estimated light intensity values and the pre-processed measurements as input, and evaluates the accuracy of the reported sensor measurements. In this example, the sensing application choose to use the similarity-based accuracy metric, which is defined as the KL divergence between the distributions of the measurement and the estimated state, as introduced in Section 3.4.2. In this case, the light intensity measurements that share more mutual information with the estimated states are considered to be more accurate. Therefore in practice, the implementation of this layer mainly depends on the accuracy metric specified by the sensing application, which maps the reported measurements to their accuracy given the states estimated by the previous layer.

Accuracy indexing layer: The output of the previous accuracy estimation layer is the accuracy value of each measurement reported by sensor systems sn_1 and sn_2. This accuracy indexing layer further indexes the computed accuracy values for individual sensor systems, according to the attributes provided by the sensing application. In this example, the application has specified the attribute *location* to index accuracy. Therefore for each sensor system (sn_1 and sn_2 in this example), this layer aggregates the accuracy of measurements observed in the same locations, and uses the aggregated accuracy at all locations to build accuracy indices of the systems. This layer then outputs the built accuracy indices, based on which the application can select the desired system (sn_1 or sn_2) to task.

In practice, the implementation of the accuracy indexing layer depends largely on the scenarios. For instance, in this example of environmental monitoring, this accuracy indexing layer does not need to include the accuracy interpolation step (discussed in Section 3.5.2), since the evaluated measurement accuracy is sufficient to build the complete

accuracy indices over locations for all systems. However, consider another positioning scenario where the sensing application would like to estimate the accuracy of coexisting positioning systems at different locations. In this case the accuracy interpolation step may be necessary, because typically we do not know the accuracy of the positioning systems at locations that haven't been visited by the users. Moreover, in this scenario, the attribute that used to index accuracy is the *ground truth*, whose values are unknown when the position measurements are observed. As discussed in Section 3.5.1, for this special case the proposed framework can be implemented by combining the state estimation, accuracy estimation and accuracy indexing layers, and addressing the joint problem of accuracy estimation and indexing. In Chapter 5, we will present novel techniques to address this joint problem in more detail.

3.7 Discussion

In this chapter, we have proposed a general accuracy estimation framework, which can be applied in various sensing scenarios. The proposed framework accepts queries from the sensing application, pulls data from the underlying sensor systems and estimates their accuracy based on the received sensor measurements. The proposed framework contains four layers: pre-processing, state estimation, accuracy estimation and accuracy indexing, where the accuracy estimation problem is addressed step by step. We have shown that raw sensor measurements are firstly cleaned in the pre-processing layer to have the same granularity and probabilistic form, and the states are estimated with measurements from all systems and available priors in the state estimation layer. The accuracy of the sensor measurements is then evaluated with respect to the estimated states in the accuracy estimation layer, and further aggregated to build the accuracy indices according to the requirements of the sensing application in the accuracy indexing layer.

We have shown that the layers in the proposed framework can be implemented in different ways. Particularly, for the state estimation layer we have created a taxonomy of state estimation approaches, ranging from the simple voting approach, to more sophisticated inference and learning techniques. We have explained the high level ideas of those state estimation approaches, and shown that inference and learning can be further divided into static and dynamic variants, depending on whether the dynamics of the monitored states are taken into account. We have also shown that the prior knowledge on the monitored states can be incorporated into both the inference and learning techniques, in both static

and dynamic cases. For the accuracy estimation layer, we have proposed two distinct accuracy metrics, a proximity-based and a similarity-based accuracy metric, and shown that they can be used in sensing applications with different types of accuracy requirements.

In practice, of course, the accuracy estimation problem can take various forms in different contexts, and the proposed framework provides a general solution to estimate the accuracy of multiple coexisting sensor systems. It relies on the fact that the accuracy of sensor systems depends on the accuracy of the measurements they report, while the accuracy of a sensor measurement is a function of the monitored state. Therefore the key idea of the proposed framework is to firstly infer the monitored states with all the available information, and then estimate the accuracy accordingly. In the next chapter, we will present a novel inference-based accuracy estimation approach which implements the proposed framework, and show how this approach works in practice in both discrete and continuous cases.

Chapter 4

Inference-based Approach for Accuracy Estimation

In this chapter, we present the inference-based approach for accuracy estimation, which implements the general framework proposed in the previous chapter. We follow the notations defined in Section 1.4 and consider the general scenario where M sensor systems $sn_1, ..., sn_M$ are monitoring the physical signal x_t, e.g. the temperature of a room or the location of a user. In this context, x_t is assumed to be a stochastic process, which evolves over discrete timestamps $t = 1 : T$. At any given timestamp t, we assume that after the pre-processing layer, each sensor system sn_m, $1 \leq m \leq M$, generates a probabilistic measurement z_t^m of the state x_t, which is a random variable with probability distribution $p(z_t^m)$. Both the monitored states and sensor measurements are assumed to be defined on the sample space Ω. We also assume that at certain timestamps, some prior information about the distribution of the states x_t may be possessed, which is referred to as prior state distributions or priors, as defined in Section 1.4.

As discussed in the previous chapter, the accuracy estimation framework contains four layers, namely a) the pre-processing layer, b) the state estimation layer, c) the accuracy estimation layer, and d) the accuracy indexing layer. The proposed inference-based approach follows this framework, and casts the accuracy estimation problem into that of a state inference problem. Concretely, the raw sensor measurements are firstly cleaned through the pre-processing layer. In the state estimation layer, the proposed approach models the real signals and the sensor measurements with probabilistic models, and estimates the states with the observed measurements, priors and known model parameters (in this chapter we always assume the model parameters are known a prior, and we will explain how they are initialised with the observed data in the next chapter). With the estimated states, the accuracy of the measurements is evaluated in the accuracy estimation layer, and then aggregated to build the accuracy indices in the accuracy indexing layer. Therefore in this chapter, we

focus on the state estimation layer and assume the other layers are implemented by existing techniques.

Particularly, we consider two different inference techniques: the *static inference* and the *dynamic inference*, depending on whether we exploit the temporal correlations between the latent states. Static inference does not consider the correlations between states, and its estimates of the states are only based on the data observed at single timestamps. On the contrary, dynamic inference treats the latent states as a dynamic process with temporal correlations, and estimates the hidden state with the full set of observed data, including measurements made in both the past and future. The technical contributions of this chapter are as follows:

- We demonstrate how to model the monitored states and observed sensor measurements with probabilistic graphical models. We show how the proposed inference-based approach can augment the existing models to accommodate the probabilistic measurements reported by multiple sensor systems, and the prior knowledge on states.

- We propose two different variants of the inference-based approach for the augmented models, the static inference and dynamic inference. We show that the proposed static and dynamic inference can work in both the discrete and continuous cases, where the latent states are estimated with the observed probabilistic sensor measurements and available priors.

- We also show that the quality of the inferred states can significantly influence the quality of accuracy estimation. We demonstrate that under the proximity-based accuracy metric, the difference between the estimated accuracy and the real accuracy can be bounded by the distance between the estimated state and the real state.

The rest of this chapter is organised as follows. Section 4.1 presents the proposed inference-based approach in the discrete case, where we show the static and dynamic inference, and highlight the novel contributions. Section 4.2 explains the proposed approach in the continuous case, where we show how to modify the existing Kalman smoother to accept probabilistic measurements and priors. Section 4.3 discusses how the quality of state estimation may influence that of accuracy estimation, and Section 4.4 concludes this chapter. The two motivating examples introduced in Section 1.3, the indoor positioning scenario and the environmental monitoring scenario, will be intensively used throughout this chapter.

4.1 Inference-based Approach for Discrete State Space

We first consider the case where the physical phenomenon monitored by the sensor systems is discrete, and assume the sample space of the states Ω is a finite discrete set. Without loss of generality, we assume that the sensor measurements are discrete as well; in practice continuous measurements can be always discretised according to the discrete state space Ω. In this section we use the indoor positioning scenario introduced in Section 1.3.1 as the running example, where multiple indoor positioning systems are providing measurements on the locations of a user, and the proposed techniques can be applied to more general cases.

Formally, we assume that the indoor environment can be represented as a finite set L, with N discrete locations $l_1, ..., l_N$, e.g. different rooms or corridor segments. At a given timestamp t, a probabilistic measurement z_t^m reported by the m-th sensor system can be represented as a vector of probabilities $[z_t^m(1), ..., z_t^m(N)]$, where $z_t^m(j)$ is the belief of the system that the user is in location l_j, $1 \leq j \leq N$. We use the same representation for the estimated state $\hat{x}_t$ and the priors ρ_t, where the distribution of $\hat{x}_t$ is represented as $[\hat{x}_t(1), ..., \hat{x}_t(N)]$, and $[\rho_t(1), ..., \rho_t(N)]$ for ρ_t. We also assume that the prior ρ_t exists at each timestamp: if prior information on the state x_t is available at time t, the distribution of ρ_t can carry such knowledge; otherwise ρ_t is uniformly distributed. Finally, we assume that the initial state can be known exactly, e.g. in this case the initial position of the user can be obtained from the card swipe at the main entrance. The rest of this section will explain in detail the proposed static and dynamic inference techniques for the discrete case.

4.1.1 Static inference

Static inference ignores any temporal correlations between the monitored states, and assumes the latent states at different timestamps are *independent*. Therefore conceptually, static inference operates on "slices" of the observed data: at a given timestamp t, probabilistic measurements from the M coexisting sensor systems and prior knowledge on the state are combined to estimate the unknown x_t. To achieve this, the proposed static inference approach augments the classic Bayesian models, and develops new inference algorithms accordingly. In the following text, we first assume there is only one sensor system reporting deterministic measurements, which is trivial and can be addressed by the existing techniques. Then we show how the proposed approach can take probabilistic measurements into account. Finally, we explain how prior state distributions and measurements from multiple coexisting sensor systems can be incorporated to help approximate the states in our approach.

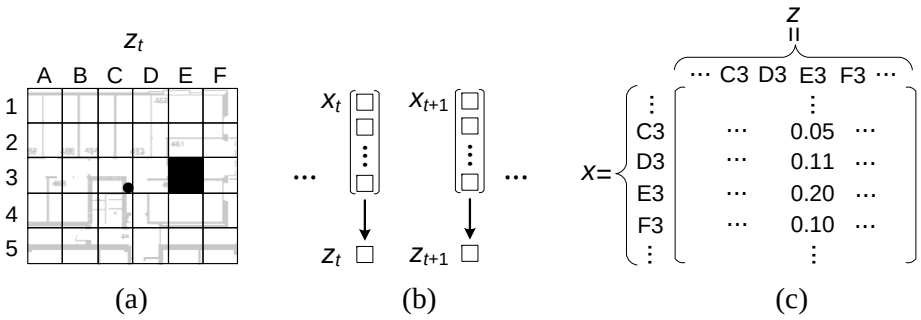


Figure 4.1: (a) A deterministic sensor measurement z_t at time t, which is a single location E3, represented by the black block. The black dot indicates the actual position of the user. (b) The model with deterministic measurements used in static inference. (c) The emission probabilities used in this model. For instance, $b_{\text{C3}}(\text{E3})$ is 0.05, which means given the state is C3, the system has 0.05 probability of reporting a measurement E3.

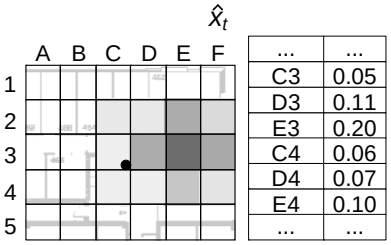


Figure 4.2: The distribution of the estimated state $\hat{x}_t$ computed by static inference on the naive model shown in Figure 4.1, where darker blocks indicate more probability mass. The distribution can also be represented as a vector of probabilities, as shown on the right.

4.1.1.1 Single sensor system with deterministic measurements

Let us first consider the simplest case where a single sensor system monitors the states and reports deterministic measurements. This case can be addressed by an existing model and inference algorithm, and in the rest of this subsection we briefly explain how they work.

Probabilistic model: Let x_t be the monitored state at timestamp t, which represents the actual location that the user is in, and z_t be the measurement reported by the sensor system, which is a single location in this case. Figure 4.1(a) shows such a deterministic measurement z_t, where the user is reported in location E3 at time t. Intuitively, z_t is correlated with the state x_t: at each timestamp t, the observed location z_t should depend on the real location x_t. Such conditional dependency can be captured by the well-known Bayesian network [29], which is a special type of probabilistic graphical models.

For a set of random variables V, a Bayesian network is a pair $B = \langle G, \theta \rangle$. G is a directed acyclic graph with nodes representing the random variables V, and edges showing the dependencies between the variables. θ represents the parameters of the Bayesian

network, where $\theta(v_i) = p(v_i|Parents(v_i))$ is the conditional probability distribution of variable v_i given the set of its parent variables $Parents(v_i)$. Therefore, the Bayesian network B describes the joint probability distribution of the random variables in V:

$$p(V) = \prod_{i=1}^{|V|} p(v_i|Parents(v_i)) = \prod_{i=1}^{|V|} \theta(v_i) \tag{4.1}$$

In our context, we consider a special form of Bayesian network, as shown in Figure 4.1(b). The unknown state x_t (the actual position of the user at time t) is modelled as a random variable with probability distribution $p(x_t)$, which in this case is a vector of probabilities $[x_t(1), ..., x_t(N)]$. The sensor measurement z_t is also considered as a random variable, but at time t the sensor observation is the event that z_t is assigned to a single (deterministic) location l_k, $1 \leq k \leq N$, as shown in Figure 4.1(a). The conditional dependency between the variables x_t and z_t is captured by the directed edge from the node x_t to z_t, as shown in Figure 4.1(b). Particularly, the conditional probability of observing a location l_k given the fact that the actual position of the user is l_j, is referred to as the *emission probability*. In practice, we often assume the emission probabilities are time-independent; we denote them as $b_j(k)$, $1 \leq j, k \leq N$, and regard them as known model parameters. Figure 4.1(c) shows the emission probabilities of this model, which from a probability matrix: for instance, $b_{\text{C3}}(\text{E3})$ is 0.05, which means given the state is in location C3, the system has 0.05 probability of reporting a measurement E3.

Inference algorithm: Given the probabilistic model and the sensor measurements, our task is to compute the estimated state $\hat{x}_t$, which approximates the actual state x_t. As stated in Chapter 3, we assume that the distribution of $\hat{x}_t$ is given by the posterior state distribution, which is a vector of probabilities $[\hat{x}_t(1), ..., \hat{x}_t(N)]$. Assuming at time t the sensor system reports that the user is at location l_k, then we have: $\hat{x}_t(j) = P(x_t = l_j|z_t)$, which is the probability that the user is actually in location l_j given measurement z_t. To avoid confusion, we use $P(\cdot)$ to represent the probability that an event happens, while $p(\cdot)$ is the probability distribution of a random variable. According to the Bayes rule, the probability $\hat{x}_t(j)$ can be evaluated as:

$$\hat{x}_t(j) = \frac{P(x_t = l_j, z_t)}{P(z_t)} = \frac{P(z_t|x_t = l_j)P(x_t = l_j)}{P(z_t)} \tag{4.2}$$

where $P(x_t = l_j)$ is the prior belief that the user is at location l_j. We typically assume a uniform distribution in the absence of other information. Given that z_t is l_k, $P(z_t|x_t = l_j)$ is in fact the emission probability $b_j(k)$, which is the likelihood of reporting the user at location l_k while she is actually at location l_j. The denominator $P(z_t)$ is the marginal probability

of the measurement z_t (l_k in this case), which can be evaluated via the marginalisation process, i.e. by summing out the latent variable x_t from the joint distribution:

$$\begin{aligned} P(z_t) = \sum_{j=1}^{N} P(x_t = l_j, z_t) &= \sum_{j=1}^{N} P(x_t = l_j) P(z_t | x_t = l_j) \\ &= \sum_{j=1}^{N} P(x_t = l_j) b_j(k) \end{aligned} \tag{4.3}$$

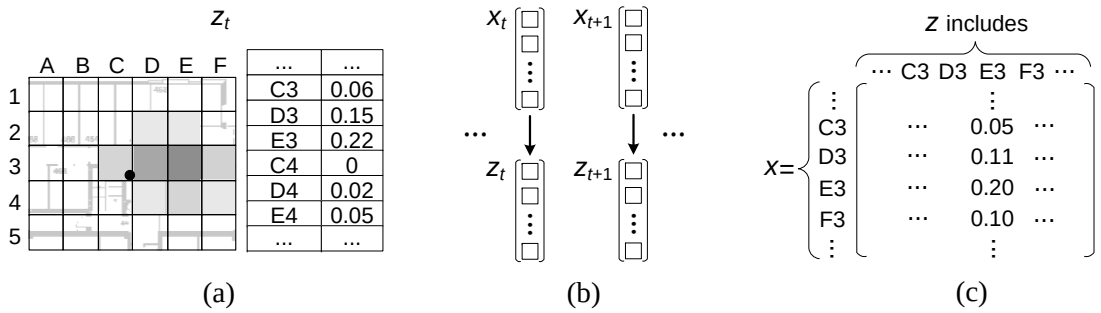
Complexity: Equations (4.2) and (4.3) have already given an inference algorithm for this model, which enumerates all possible locations in L for x_t. In fact, they depict the key idea of the widely used variable elimination algorithm [29] for Bayesian models. Given the structure of our model, the complexity of this inference algorithm is $O(TN)$, where T is the total number of timestamps and N is the size of L, since at each timestamp we only need to enumerate N possible locations.

Example: Given the measurement z_t (a single location E3) and the model parameters (emission probabilities) shown in Figures 4.1(a) and (c), Figure 4.2 shows the distribution of the estimated state $\hat{x}_t$. In fact, the distribution of $\hat{x}_t$ is computed as in Equation (4.2). For example, according to Equation (4.2), the probability that $\hat{x}_t$ is location E3 (i.e. l_j is E3 in this case) is given by the product of the emission probaiblity $b_{\text{E3}}(\text{E3})$ (0.20 in this example) and the prior probability that the state is in location E3, which is assumed to be uniform. From the result in Figure 4.2, we can see that the majority of the probability mass of $\hat{x}_t$ is concentrated around the reported location E3, since in this case the only information we possess is the reported location E3 and the model parameters $b_j(k)$, which are shown in Figure 4.1(c).

4.1.1.2 Single sensor systems with probabilistic measurements

Now we consider the case that the measurements of a sensor system are *probabilistic*, a case not captured by the existing techniques discussed in the previous subsection.

This is common in practice, since sensor systems usually pair the estimated values of the monitored states with a certain measure of uncertainty, such as confidence intervals or error ellipsoids, to indicate the belief about the reported measurements. For example in the indoor positioning scenario, a positioning system typically provides the estimated positions together with the areas in which the user is likely to lie in. As stated at the beginning of this chapter, the probabilistic measurement z_t at timestamp t is a random variable with distribution $p(z_t)$ over the locations L. Unlike the deterministic case where the sensor system only reports a single location l_k, $1 \leq k \leq N$ at time t, in this case the



(a) (b) (c)

Figure 4.3: (a) A probabilistic sensor measurement z_t at time t, which is a distribution over locations represented by the grey blocks (darker blocks indicate more probability mass). The black dot indicates the actual position of the user. z_t can also be represented as a vector of probabilities (on the right), each of which is the belief of the system that the user is in that particular location. (b) The model with probabilistic measurements used in static inference. (c) The emission probabilities of this model have a different meaning. For instance, now $b_{\text{C3}}(\text{E3})$ means the expected probability that a measurement z_t has location E3 given the state is C3, which is 0.05.

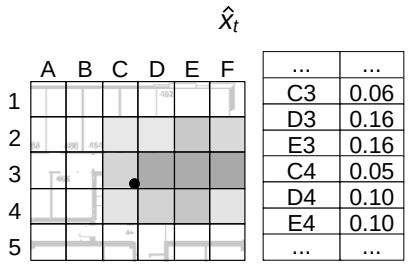


Figure 4.4: The estimated state computed by static inference on the model with probabilistic observations (shown in Figure 4.3).

sensor observation is the whole distribution $p(z_t)$, as shown in Figure 4.3(a). This cannot be described by the simple Bayesian models discussed in the previous section, and now we show how we address this with a new model.

Probabilistic model: In the discrete case, the probability distribution of a random variable can be represented as a vector of probabilities. Therefore, to accommodate probabilistic sensor measurements, we augment the simple model to accept vector observations: at time t, a measurement z_t is a probability vector $[z_t(1), ..., z_t(N)]$, where $z_t(k)$ indicates the belief of the sensor system that the user is in location l_k. In other words, rather than observing a single location l_k at a time, the augmented model observes N locations, l_1, ..., l_N, associated with the probabilities (or weights) $z_t(1)$, ..., $z_t(N)$. Figure 4.3(a) shows such a vector observation, where each entry is a pair $\langle \text{Loc}, p \rangle$, e.g. $\langle \text{D3}, 0.15 \rangle$ indicates that the system believes the user has 0.15 probability of being in location D3 at time t. Figure 4.3(b) illustrates the augmented model, where the state x_t and x_{t+1} emit probabilistic measurements

z_t and z_{t+1} respectively.

Inference algorithm: For this augmented model, now we introduce a new inference algorithm which is able to consume this type of observations. In this case, the distribution of the estimated state $\hat{x}_t$, i.e. the posterior state distribution, is given by:

$$\hat{x}_t(j) = \frac{P(x_t = l_j, z_t)}{P(z_t)} = \frac{P(z_t|x_t = l_j)P(x_t = l_j)}{P(z_t)} \tag{4.4}$$

Unlike Equation (4.2), note that here $P(z_t|x_t = l_j)$ is the probability of observing the *probabilistic* measurement z_t, i.e. the vector, given the fact that the user is actually in location l_j. $P(z_t)$ is the marginal probability that the probabilistic measurement z_t is observed.

To compute $P(z_t|x_t = l_j)$, consider the extreme case where at time t the measurement z_t is *missing*, or *unobserved*. In that case, since the variable z_t becomes hidden, it is equivalent to regarding z_t as a uniformly distributed variable. Then given the fact that x_t is l_j, the probability $P(z_t|x_t = l_j)$ can be computed by summing over all possible locations l_k for the variable z_t:

$$P(z_t|x_t = l_j) = \frac{1}{N}\sum_{k=1}^{N} P(z_t = l_k|x_t = l_j) \tag{4.5}$$

where $\frac{1}{N}$ is the normalising constant to make $P(z_t|x_t = l_j)$ a valid probability measure. It can also be viewed as the *weight* of each location l_k. Since z_t is missing, the weights are the same (i.e. $\frac{1}{N}$) for all the locations l_k. We extend this to a more general case where the observed probabilities $[z_t(1), ..., z_t(N)]$ are used as the weights:

$$P(z_t|x_t = l_j) = \sum_{k=1}^{N} z_t(k)P(z_t = l_k|x_t = l_j) \tag{4.6}$$

Here the term $P(z_t = l_k|x_t = l_j)$ is the expected probability of having location l_k in measurement z_t given the user is actually at location l_j. We also refer to this probability as the *emission probability* $b_j(k)$, and assume it to be a known model parameter. However, note that the emission probability in this augmented model has a different meaning from that in the deterministic case discussed previously. In the deterministic case where the observation is a single location, the emission probability $b_j(k)$ determines the *likelihood of making an observation* l_k given the state l_j, as shown in Figure 4.1(c); however in this case, $b_j(k)$ indicates the *expected probability* allocated to l_k in a probabilistic measurement given the state l_j. For instance, Figure 4.3(c) shows the emission probabilities used in our context. Here the emission probability $b_{\text{C3}}(\text{E3})$ represents the expected probability that location E3 is reported in a measurement given the state is C3.

As shown in Equation (4.6), the likelihood of making an observation is determined by both the emission probabilities and the probabilistic measurement observed. Feeding Equation (4.6) into Equation (4.4), we have:

$$\hat{x}_t(j) = \frac{P(x_t = l_j) \sum_{k=1}^{N} z_t(k) b_j(k)}{P(z_t)} \tag{4.7}$$

where $P(x_t = l_j)$ is again the prior belief. The marginal probability $P(z_t)$ can be computed by enumerating all possible locations for the state x_t as shown in the deterministic case:

$$\begin{aligned} P(z_t) &= \sum_{j=1}^{N} P(x_t = l_j, z_t) \\ &= \sum_{j=1}^{N} \Big[P(x_t = l_j) \sum_{k=1}^{N} z_t(k) b_j(k) \Big] \end{aligned} \tag{4.8}$$

Complexity: Therefore, to compute the distribution of the estimated state $\hat{x}_t$, we need to enumerate all possible locations *twice*: one is for the latent state x_t, and the other is for the probabilistic measurement z_t. Note that the second enumeration is weighed by the probability distribution of z_t, which carries the knowledge of the observed probabilistic measurement. This means the complexity of our algorithm is $O(TN^2)$ due to the presence of probabilistic measurements. However in practice, the probabilistic measurements can be sparse (only a few locations have non-zero probabilities), and with some optimisations our algorithm is not prohibitively expensive.

Example: Consider the probabilistic measurement z_t shown in Figure 4.3(a), which is a probability distribution over 30 discrete locations. The model parameters, i.e. emission probabilities are shown in Figure 4.3(c). For instance, the value 0.05 represents the probability of having the location E3 in the probabilistic measurement given that the actual location is C3. Figure 4.4 shows the posterior state distribution of $\hat{x}_t$ after observing the probabilistic measurement z_t. The probability $\hat{x}_t(j)$ is computed according to Equation (4.7). For example, to compute the posterior probability that the state in location E3, we need to compute the summation of products: $z_t(\text{A1}) b_{\text{E3}}(\text{A1}) + \ldots + z_t(\text{F5}) b_{\text{E3}}(\text{F5})$ for all locations (A1 to F5), while the prior distribution $P(x_t = \text{E3})$ and marginal probability $P(z_t)$ can be viewed as normalising factors.

4.1.1.3 Multiple sensor systems with prior state distributions

The previous subsection shows how our approach can incorporate probabilistic observations by augmenting the existing model and inference algorithm. Now we consider an even

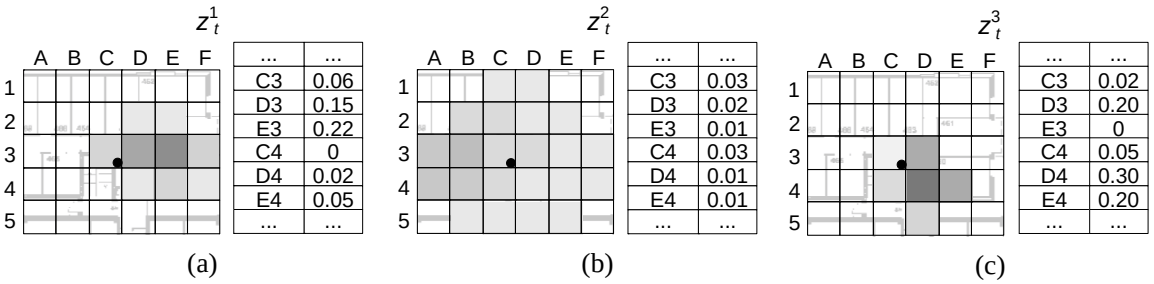
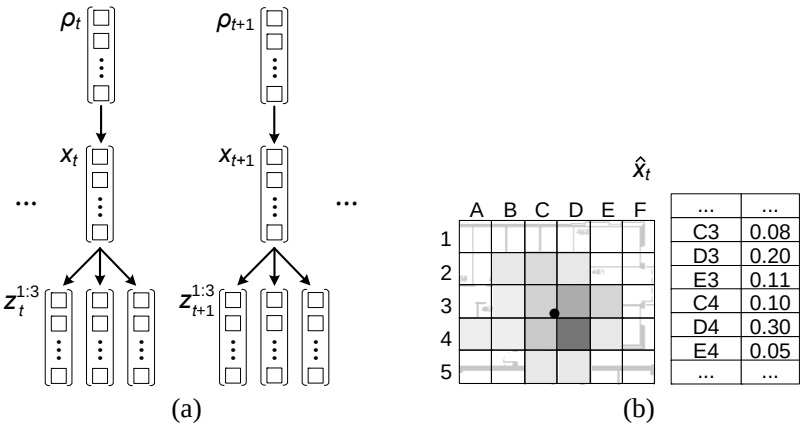


Figure 4.5: At time t, the probabilistic sensor measurements z_t^1, z_t^2 and z_t^3 reported by three sensor systems. Each of them is a distribution over locations (vector of probabilities), visualised by the grey blocks where darker blocks indicate more probability mass. Note that z_t^1 is the measurement considered in the previous subsection, and the black dot indicates the actual position of the user.

more complex case, where multiple sensor systems report probabilistic measurements, and prior knowledge on state is available. In this case, we assume at time t, M sensor systems sn_1, ..., sn_M report probabilistic measurements z_t^1, ..., z_t^M, and a prior state distribution ρ_t is available, e.g. from the calendar information of the user. Figure 4.5 shows an example where three probabilistic measurements are reported by the positioning systems, each of which is a vector of probabilities.

Probabilistic model: In the presence of multiple sensor measurements, the model proposed in the previous subsection needs to be further augmented. Intuitively, all the measurements should depend on the state, e.g. the location measurements in Figure 4.5 are noisy observations of the actual user position (the black dot). Here we make an important assumption, that given the state x_t, the sensor measurements z_t^m are *conditionally independent*. For instance, given the user is actually in location l_j at time t, the observed location measurements z_t^1, ..., z_t^M are independent of each other. This assumption is valid when there is no external noise and interference to the sensor systems, or in the cases where it can be ignored. Incorporation of the prior ρ_t is straightforward: from the Bayesian point of view, ρ_t is the prior information on the state x_t before any measurement is observed. Figure 4.6(a) shows the augmented model used in this case, where at a timestamp there are multiple probabilistic measurements, and the priors are represented as the parent nodes of the states.

Inference algorithm: Given multiple probabilistic sensor measurements and prior on state, our inference approach uses the assumption of conditional independence discussed above,



(a) (b)

Figure 4.6: (a) The model used in the static case where probabilistic measurements from multiple sensor systems and prior state distributions are incorporated. (b) The estimated state $\hat{x}_t$ computed by static inference on this model. Comparing with that in the previous case (shown in Figure 4.4), since here we have multiple measurements at a timestamp that can validate each other, more probability mass of $\hat{x}_T$ is concentrated in locations that are close to the ground truth.

and estimates the posterior state distribution as follows:

$$\hat{x}_t(j) = \frac{P(x_t = l_j, \rho_t, z_t^{1:M})}{P(z_t^{1:M}, \rho_t)} = \frac{P(x_t = l_j, \rho_t) \prod_{m=1}^{M} P(z_t^m | x_t = l_j, \rho_t)}{P(z_t^{1:M}, \rho_t)} \tag{4.9}$$

where the term $\prod_{m=1}^{M} P(z_t^m | x_t = l_j, \rho_t)$ computes the likelihood of observing the M probabilistic sensor measurements, since they are independent conditioned on the state. Note that $P(z_t^m | x_t = l_j, \rho_t)$ is actually $P(z_t^m | x_t = l_j)$, since the measurements are assumed to be independent of the prior given the state. Using the marginalisation approach introduced in Equation (4.6), $\hat{x}_t(j)$ can be computed as:

$$\hat{x}_t(j) = \frac{P(x_t = l_j, \rho_t) \prod_{m=1}^{M} \sum_{k=1}^{N} z_t^m(k) P(z_t^m = l_k | x_t = l_j)}{P(z_t^{1:M}, \rho_t)} \tag{4.10}$$

where $z_t^m(k)$ is the k-th probability from the probabilistic sensor measurement z_t^m. Recall that $P(z_t^m = l_k | x_t = l_j)$ is the expected probability of having location l_k in measurement z_t^m reported by system sn_m given that the user is at location l_j. As in the previous case, we also refer to it as the *emission probability*, and denote it as $b_j^m(k)$. Note that here the emission probabilities $b_j^m(k)$ are defined for each system sn_m independently, and are assumed to be model parameters known a priori. $P(x_t = l_j, \rho_t)$ represents the prior belief on state carried by the prior ρ_t, i.e. $P(x_t = l_j, \rho_t) = \rho_t(j)$, where $\rho_t(j)$ is the j-th probability in

the distribution of ρ_t. Plugging these in Equation (4.10), the distribution of estimated state becomes:

$$\hat{x}_t(j) = \frac{\rho_t(j) \prod_{m=1}^{M} \sum_{k=1}^{N} z_t^m(k) b_j^m(k)}{P(z_t^{1:M}, \rho_t)} \tag{4.11}$$

where the normalising factor $P(z_t^{1:M}, \rho_t)$ can be computed in a similar way as in Equation (4.8), but with multiple probabilistic sensor measurements and priors, as shown in Equation (4.11).

Complexity: From Equation (4.11) we can see that the complexity of this algorithm is now $O(TMN^2)$, since we have to marginalise the probabilistic observations from M systems. Note that in practice M is typically much smaller than N (say 3$\sim$5 coexisting systems), and therefore this factor will not increase the cost of our algorithm significantly.

Example: Consider the probabilistic measurements z_t^1, z_t^2 and z_t^3 received from three sensor systems, as shown in Figure 4.5. We assume the emission probabilities are the same as in Section 4.1.1.2, as shown in Figure 4.3(c). For simplicity we use uniform prior in this example. Figure 4.6(b) shows the state estimated with the multiple probabilistic sensor measurements. Comparing with that in the previous case where only one probabilistic measurement z_t^1 is considered (shown in Figure 4.4, here to compute the posterior state distribution $\hat{x}(j)$, we need to compute the summation of products for each of the three sensor systems, and multiply the results together as shown in Equation (4.11). For instance to evaluate x_t(E3), we compute the following three summations of products: z_t^1(A1)b_{E3}^1(A1)+ ... +z_t^1(F5)b_{E3}^1(F5), z_t^2(A1)b_{E3}^2(A1)+ ... +z_t^2(F5)b_{E3}^2(F5), and z_t^3(A1)b_{E3}^3(A1)+ ... +z_t^3(F5)b_{E3}^3(F5) for all three systems, and then multiply them together. We also need to multiply the prior state distribution (if it is not uniform) to take the prior information on the state into account. As shown in Figure 4.4(b), the probability mass of the estimated state here is concentrated to locations that are closer to the ground truth (D3 and D4), since now we have multiple measurements that are fused to estimate the true user location.

4.1.2 Dynamic inference

We now present the dynamic inference-based approach in the discrete case. Unlike the static inference, dynamic inference models the states x_1, ..., x_T as a random process. As stated in Section 3.3.2.2, we assume that the temporal correlations between the states are *Markovian*, i.e. $p(x_t|x_{1:t-1}) = p(x_t|x_{t-1})$. The Markovian assumption indicates that the current state x_t depends only on the immediately previous state x_{t-1}, and is independent

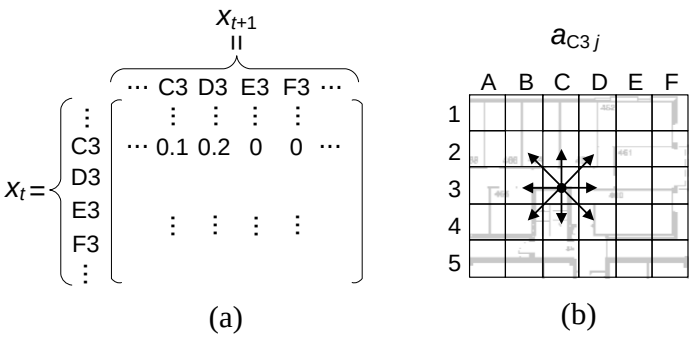


Figure 4.7: (a) The transition probabilities a_{ij}. (b) For a given location C3, the transition probabilities a_{C3j} indicates the probabilities of moving from C3 to another locations.

of the states further before. In the positioning scenario, it requires that the current position of the user only depends on her previous position, which is reasonable. For simplicity, we also assume that the distribution $p(x_t|x_{t-1})$ is independent of time t, i.e. the Markovian process x_1, ..., x_T is *stationary*. For given locations l_i and l_j, $1 \leq i, j \leq N$, the probability $P(x_t = l_j|x_{t-1} = l_i)$ is referred to as the *transition probability*, denoted as a_{ij}, which indicates the probability of transiting from one location l_i to another l_j. In the proposed approach, the transition probabilities a_{ij} are regarded as model parameters, and assumed to be known a priori. Figure 4.7(a) shows an example of the transition probabilities, which is a probability matrix. The probability 0.2 indicates the likelihood of transiting from location C3 to D3 at time t and $t + 1$. Figure 4.7(b) shows the transition probabilities $a_{C3\,j}$ for location C3, where we can see that in this case, given the current state is C3, the user is can only transit to the nearby locations.

We also assume that the proposed approach works in an off-line manner, which performs inference only when data at all timestamps $1 : T$ have been received. We follow the structure as in static inference, and firstly explain the simple case with deterministic sensor measurements, which can be addressed by an existing technique. Then we show how the proposed approach can incorporate probabilistic measurements in this dynamic setting, and finally demonstrate how multiple sequences of probabilistic measurements and priors can be taken into account.

4.1.2.1 Single sensor system with deterministic measurements

We first consider the simplest case where only one sensor system reports a sequence of deterministic measurements $z_{1:T}$. This can be addressed by the existing model and inference algorithm, and we explain the key ideas here to set up the scene before introducing our approaches.

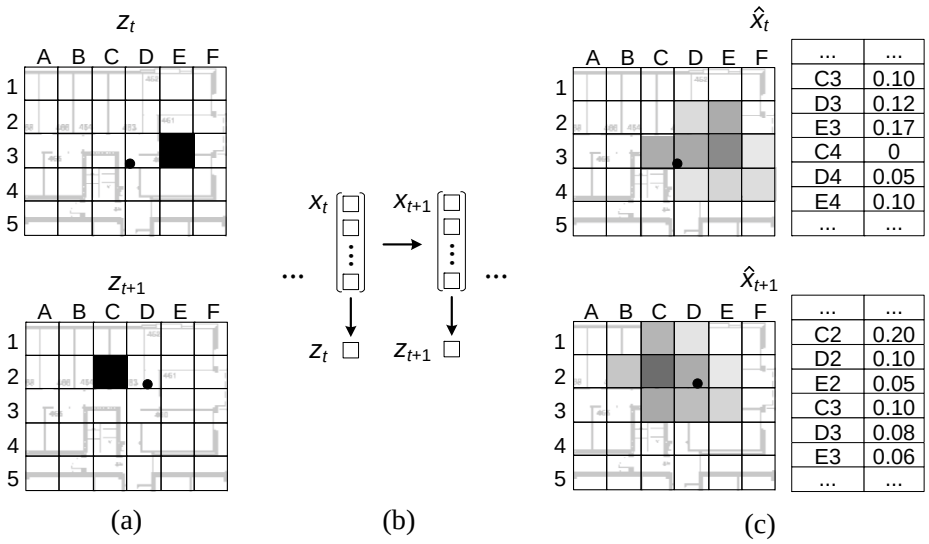


Figure 4.8: (a) The deterministic sensor measurements z_t (a single location E3) and z_{t+1} (a single location C2) observed at time t and $t+1$. The black dots indicate the actual positions of the user. (b) The model used in this case, which emits deterministic measurements and the states are correlated (the directed edge between x_t and x_{t+1}) (c) The estimated states $\hat{x}_t$ and $\hat{x}_{t+1}$ computed by the existing forward-backward algorithm [2]. Comparing $\hat{x}_t$ with that in static case (shown in Figure 4.2), the location D3 (closer to the ground truth) is assigned with more probability mass, while location E3 (far away from the ground truth) is less, indicating that the state estimate computed by dynamic inference is better even with deterministic observations.

Probabilistic model: In this case, at a given time t the measurement z_t reported by the positioning system is a single location l_k. Figure 4.8(a) shows an example where at time t and $t+1$, the user is moving from one discrete location D3 to another D2, while the measurements observed are locations E3 and C2. In this dynamic case, the states at time t and $t+1$ are correlated, as shown in Figure 4.8(b). In fact, this model is the classic hidden Markov model (HMM), which is a special case of state space models.

Inference algorithm: As discussed in the background chapter, the inference task in HMMs can be addressed by the widely-used forward-backward algorithm [2]. In our particular case, the distribution of the estimated state $\hat{x}_t$ is given by:

$$\hat{x}_t(j) = \frac{P(x_t = l_j, z_{1:T})}{P(z_{1:T})} = \frac{P(x_t = l_j, z_{1:t})P(z_{t+1:T}|x_t = l_j)}{P(z_{1:T})} \tag{4.12}$$

based on the Markovian assumption and the fact that measurements at different timestamps are independent given the states. In the forward-backward algorithm, both terms $P(x_t = l_j, z_{1:t})$ and $P(z_{t+1:T}|x_t = l_j)$ are computed iteratively. They are denoted as the *forward* and *backward* variables, which are defined as follows (assuming that the measurements z_t,

z_{t+1} observed at time t and $t+1$ are locations l_k and $l_{k'}$ respectively):

$$\alpha_t(j) = P(x_t = l_j, z_{1:t}) = \Big[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\Big] b_j(k) \tag{4.13a}$$

$$\beta_t(j) = P(z_{t+1:T}|x_t = l_j) = \sum_{i=1}^{N} \Big[a_{ji} b_j(k') \beta_{t+1}(i)\Big] \tag{4.13b}$$

where a_{ij} is the transition probability of the user moving from location l_i to l_j, and $b_j(k)$ is the emission probability indicating the likelihood of reporting location l_k when the user is at location l_j, as shown in Figure 4.1(c). Concretely, the forward variable $\alpha_t(j)$ represents the joint probability of observing all the previous measurements and landing at location l_j, while the backward variable $\beta_t(j)$ is the probability of having all future measurements given the fact that the current state is at location l_j. Then substituting Equation (4.13) into Equation (4.12), the estimated state distribution $\hat{x}_t(j)$ can be computed as:

$$\hat{x}_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \tag{4.14}$$

Note that the normalising factor $P(z_{1:T})$ in Equation (4.12) is computed by summing over all possible locations l_j the product of the forward and backward variables.

Complexity: From the definition of the variables, we can see that the complexity of this algorithm is $O(TN^2)$: at each time t there are N $\alpha_t(j)$ and $\beta_t(j)$, each of which needs to sum over N locations.

Example: Figure 4.8(a) shows an example of two deterministic measurements observed at timestamps t and $t+1$. The state transition probabilities are shown in Figure 4.7, while we assume the emission probabilities are the same as the static case (Section 4.1.1.1). Figure 4.8(c) shows the estimated states at time t and $t+1$ computed by the above dynamic inference algorithm. If we compare the estimated distribution of $\hat{x}_t$ with that in the static case (shown in Figure 4.2, we can see that the location D3 which is closer to the ground truth is assigned more probability mass, while the further location E3 has less. This shows that dynamic inference can provide better state estimations even with deterministic measurements, since it takes the sequence of measurements into account, where previous and future measurements can "correct" the current estimation of the state.

4.1.2.2 Single sensor system with probabilistic measurements

Now we consider the case where the measurements from a sensor system are probabilistic. In this case, it is not possible to model the probabilistic sensor measurements with

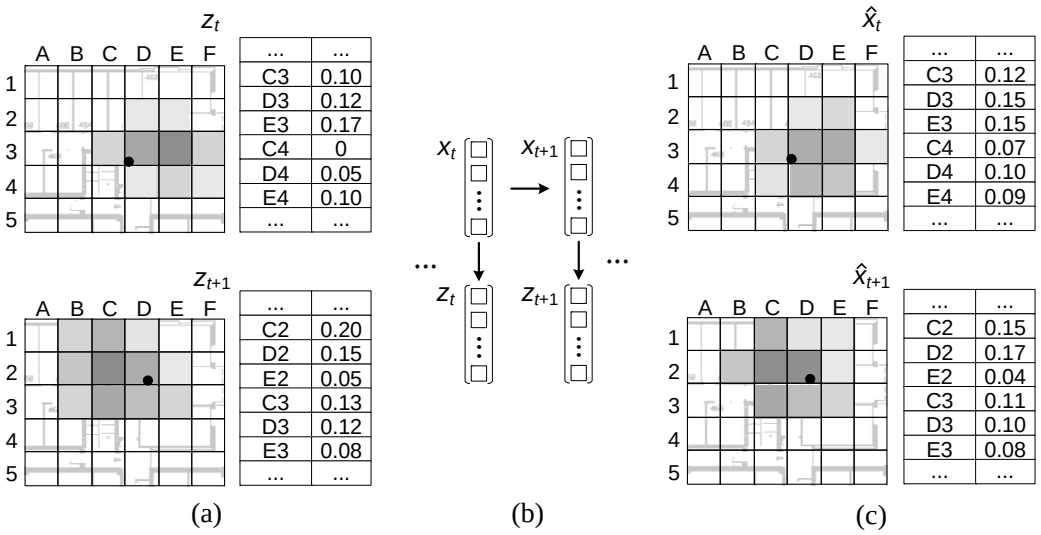


Figure 4.9: (a) The probabilistic sensor measurements z_t and z_{t+1} observed at time t and $t+1$. The black dots indicate the actual positions of the user. (b) The augmented model with probabilistic measurements and state transitions (the directed edge between x_t and x_{t+1}) used in dynamic inference. (c) The states estimated by the proposed dynamic inference algorithm. Comparing with that in the deterministic case (shown Figure 4.8(c)), the estimated states allocate more probability mass to the locations that are close the ground truth, because the probabilistic measurements considered in this case can carry more information about the user locations. The estimated state at time t is also better than that in the static case with probabilistic measurements (shown in Figure 4.4) since here we consider the sequence of measurements rather than just one.

the existing hidden Markov models, and the inference task cannot be addressed by the classic forward-backward algorithm either. In the following, we show how the proposed approach incorporates the probabilistic measurements into the model, and estimates states accordingly.

Probabilistic model: In this case, at time t the sensor observation z_t is a vector $[z_t(1), ..., z_t(N)]$, where for each location l_k, $1 \leq k \leq N$, the probability $z_t(k)$ is the belief of the system that the user is in location l_k. Figure 4.9(a) shows an example of two probabilistic sensor measurements observed at time t and $t+1$, where each of them is a vector of probabilities. We augment the standard hidden Markov model discussed in the previous subsection: instead of observing a single location l_k, at a timestamp our model emits a vector observation, which is a distribution over all the locations L, as shown in Figure 4.9(b).

Inference algorithm: Now we explain the dynamic inference algorithm proposed for this augmented model. It follows the similar forward-backward framework, but is able to incorporate the sequence of probabilistic measurements. As in the deterministic case, the

distribution of the estimated state $\hat{x}_t$ can be represented as:

$$\hat{x}_t(j) = \frac{P(x_t = l_j, z_{1:t})P(z_{t+1:T}|x_t = l_j)}{P(z_{1:T})} = \frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \tag{4.15}$$

However, here the meanings of $P(x_t = l_j, z_{1:t})$ and $P(z_{t+1:T}|x_t = l_j)$ are different, since $z_{1:t}$ and $z_{t+1:T}$ represent sequences of probabilistic measurements. Factors $\alpha_t(j)$ and $\beta_t(j)$ here are the *augmented* forward and backward variables, which take the probabilistic measurements into account. The augmented $\alpha_t(j)$ represents the probability of observing all the probabilistic measurements until time t and landing at location l_j, which is defined as:

$$\alpha_t(j) = P(x_t = l_j, z_{1:t}) = \Big[\sum_{i=1}^{N} \alpha_{t-1}(i)a_{ij}\Big] P(z_t|x_t = l_j) \tag{4.16}$$

where $\alpha_{t-1}(i)$ is the previous augmented forward variable. The term $P(z_t|x_t = l_j)$ is the probability of observing the probabilistic measurement z_t given that the state is in l_j. Recall that in the static case with probabilistic measurements (Section 4.1.1.2), $P(z_t|x_t = l_j)$ can be computed by summing over all possible locations l_k for the probabilistic measurement z_t. For notation simplicity, we denote this probability with the variable $r_t(j)$, which is defined as:

$$\begin{aligned} r_t(j) &= P(z_t|x_t = l_j) \\ &= \sum_{k=1}^{N} z_t(k)P(z_t = l_k|x_t = l_j) = \sum_{k=1}^{N} z_t(k)b_j(k) \end{aligned} \tag{4.17}$$

where $b_j(k)$ is the known emission probabilities, as defined in the static case (Section 4.1.1.2). With the above $r_t(j)$, the augmented forward and backward variables can be represented as:

$$\alpha_t(j) = \Big[\sum_{i=1}^{N} \alpha_{t-1}(i)a_{ij}\Big] r_t(j) \tag{4.18a}$$

$$\beta_t(j) = \sum_{i=1}^{N} \Big[a_{ji} r_{t+1}(i)\beta_{t+1}(i)\Big] \tag{4.18b}$$

Substituting Equation (4.18) into Equation (4.15), the distribution of the estimated state can be computed.

Complexity: If we take a closer look into Equation (4.18) in above and Equation (4.13) in the previous subsection, we can find that the augmented forward and backward variables here have a similar form to those in the deterministic case: the major difference is how the

likelihood of a sensor observation is evaluated. In the deterministic case, this likelihood can be fully determined by the emission probabilities $b_j(k)$, but in the probabilistic case considered by our approach, the likelihood of observing a measurement is the weighted sum of the emission probabilities $b_j(k)$, where the weights come from the measurements (see the definition of $r_t(j)$ in Equation (4.17)). Note that in our case it is not possible to model the likelihood of observing a probabilistic sensor measurement with any finite distribution. Therefore, due to the presence of probabilistic measurements, the complexity of our algorithm is $O(TN^3)$: the backward variable $\beta_t(j)$ needs an extra summation over all possible N locations when evaluating $r_{t+1}(i)$.

Example: Figure 4.9(a) shows an example of two probabilistic measurements observed at timestamp t and $t+1$. We assume that the transition probabilities are the same as the previous deterministic case, while the emission probabilities are identical to the static case discussed in Section 4.1.1.2. Figure 4.9(c) shows the estimated states at time t and $t+1$, computed by our dynamic inference algorithm. Comparing with those in the deterministic case (shown in Figure 4.8(c)), we can see that the states estimated by our algorithm have more probability mass assigned to locations closer to the ground truth, and thus are better. This is because the probabilistic measurements can carry more information than the deterministic ones, which are merely single locations. Similarly, the estimated state $\hat{x}_t$ is also better than that in the static case (shown in Figure 4.4), because the full sequence of probabilistic measurements are taken into account rather than just one.

4.1.2.3 Multiple sensor systems with prior state distribution

The previous subsection has explained how the proposed approach can incorporate probabilistic measurements in the dynamic case, and now we consider the final case, that multiple sensor systems report sequences of probabilistic measurements, and prior information on states is available.

Probabilistic model: We assume that M sensor systems are providing probabilistic measurements for the time period $t = 1 : T$, where $z^m_{1:T}$ is the measurement sequence reported by the m-th sensor system, $1 \leq m \leq M$. Figure 4.10 shows the probabilistic measurements reported by three coexisting positioning systems at two timestamps t and $t+1$ respectively, where measurements z^1_t and z^1_{t+1} are the same as in the previous subsection. We also assume that the measurements from multiple sensor systems are conditionally independent given the state, and the priors ρ_t exists at every timestamp t: if there is any prior knowledge on the state, the distribution of ρ_t will carry such information, otherwise ρ_t is uniformly distributed.

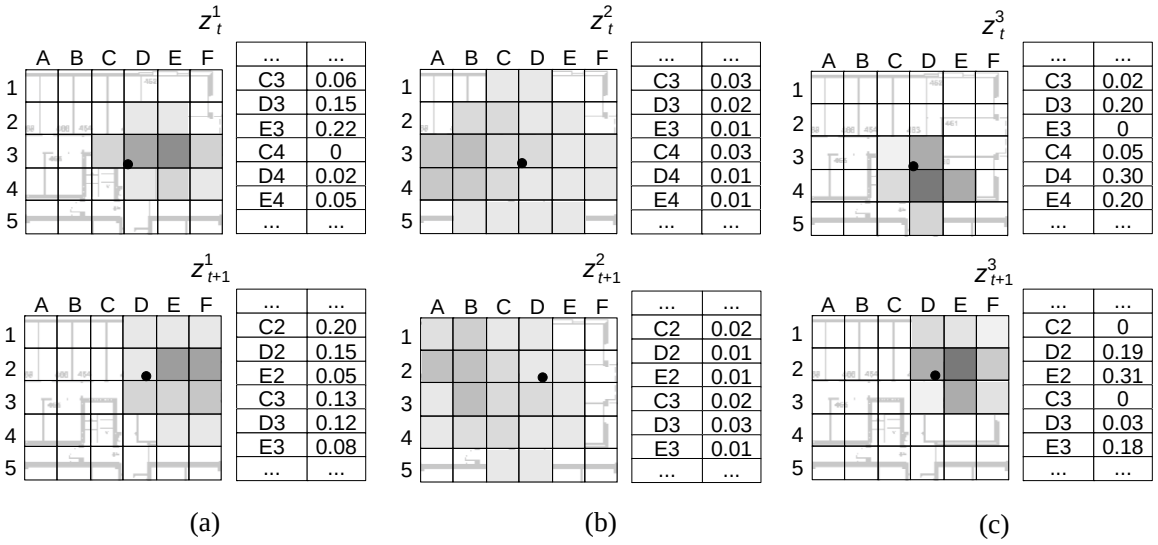


Figure 4.10: The probabilistic sensor measurements reported by three systems sn_1 (a), sn_2 (b) and sn_3 (c) at time t and $t+1$. Note that z_t^1 and z_{t+1}^1 are the same as in the previous subsection, and the black dots indicate the actual positions of the user.

We further augment the model proposed in the previous subsection, to incorporate multiple probabilistic measurements and the priors, as shown in Figure 4.11(a). Comparing with the previous model (Figure 4.9 (b)), this model now emits multiple probabilistic observations at a timestamp, and the distribution of state x_{t+1} depends on both the previous state x_t and the current prior ρ_{t+1}. Also, unlike the model used in the static case (see Figure 4.6(a) in Section 4.1.1.3), this model considers the temporal correlations between states. Now we show how to further augment the inference algorithm to take those extra elements into consideration.

Inference algorithm: The inference process on this model also follows the forward and backward framework, where the distribution of the estimated state $\hat{x}_t$ is given by:

$$\hat{x}_t(j) = \frac{P(x_t = l_j, \rho_{1:t}, z_{1:t}^{1:M}) P(z_{t+1:T}^{1:M}, \rho_{t+1:T} | x_t = l_j)}{P(z_{1:T}^{1:M})} = \frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \tag{4.19}$$

where we further augment the forward and backward variables to take the multiple sequences of probabilistic measurements and the prior into account:

$$\alpha_t(j) = \Big[\sum_{i=1}^{N} \alpha_{t-1}(i) P(x_t = l_j, \rho_t | x_{t-1} = l_i) \Big] P(z_t^{1:M} | x_t = l_j, \rho_t) \tag{4.20a}$$

$$\beta_t(j) = \sum_{i=1}^{N} \Big[P(x_{t+1} = l_i, \rho_{t+1} | x_t = l_j) P(z_{t+1}^{1:M} | x_{t+1} = l_i, \rho_{t+1}) \beta_{t+1}(i) \Big] \tag{4.20b}$$

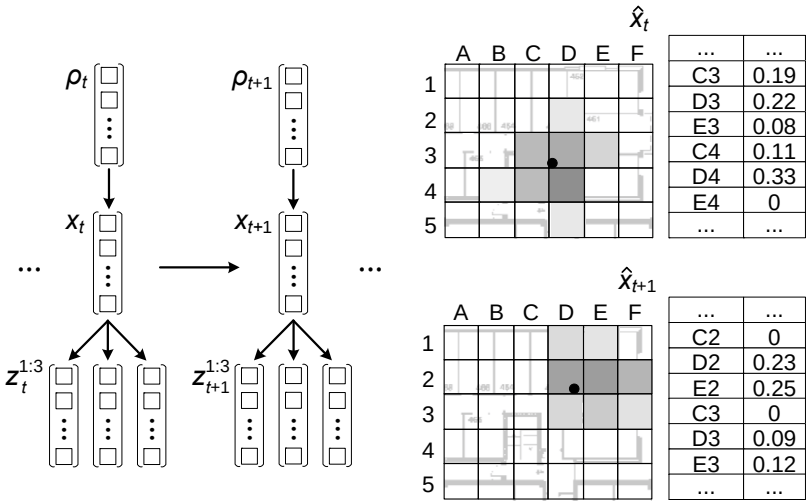


Figure 4.11: (a) The dynamic model with multiple probabilistic measurements and prior state distributions used in this case. (b) The estimated states $\hat{x}_t$ and $\hat{x}_{t+1}$ computed by dynamic inference on this model, which are better than those in the previous case with only one sequence of measurements (shown in Figure 4.9 (c)) since more information is considered. The estimated state $\hat{x}_t t$ is also superior to that in the static case (shown in Figure 4.6(b)), because the previous and future measurements can influence the estimation of current state through the model dynamics.

Here $P(z_t^{1:M}|x_t = l_j, \rho_t)$ is actually $P(z_t^{1:M}|x_t = l_j)$, which is the probability of observing the M probabilistic measurements given the state is at location l_j, since the measurements are assumed to be independent of the prior. This probability can be computed by extending the variable $r_t(j)$ in the previous subsection (see Equation (4.17)):

$$r_t(j) = P(z_t^{1:M}|x_t = l_j) = \prod_{m=1}^{M} \sum_{k=1}^{N} z_t^m(k) b_j^m(k) \tag{4.21}$$

where $z_t^m(k)$ is the k-th probability from the probabilistic measurement z_t^m reported by system sn_m, and $b_j^m(k)$ is from the emission probabilities of system sn_m, $1 \leq m \leq M$. In the presence of prior ρ_t, the term $P(x_t = l_j, \rho_t|x_{t-1} = l_i)$ in Equation (4.20) is the probability of transiting from location l_i to l_j with the knowledge of ρ_t. Since the prior ρ_t is independent of the previous state x_{t-1}, this probability can be evaluated as: $P(x_t = l_j, \rho_t|x_{t-1} = l_i) = a_{ij}\rho_t(j)$. Taking all elements into account, we now have the final

forward and backward variables for this model:

$$\alpha_t(j) = \Big[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} \rho_t(j)\Big] r_t(j) \tag{4.22a}$$

$$\beta_t(j) = \sum_{i=1}^{N} \Big[a_{ji} \rho_{t+1}(i) r_{t+1}(i) \beta_{t+1}(i)\Big] \tag{4.22b}$$

Complexity: In this case, the forward variable $\alpha_t(j)$ represents the probability of observing all the probabilistic measurements from the M systems and priors until time t, and landing at state l_j. The backward variable $\beta_t(j)$ is the probability of having all the M sequences of future probabilistic measurements and priors from time $t+1$, given that the current state is l_j. Therefore the complexity of this algorithm is $O(TMN^3)$, since we have M sequences of probabilistic measurements.

Example: Figure 4.10 shows an example of probabilistic measurements reported by three sensor systems at timestamps t and $t+1$. We use identical transition probabilities as in the previous subsection, and the emission probabilities of the three systems are the same as in the static case (Section 4.1.1.3). Figure 4.11(b) shows the estimated states at time t and $t+1$ produced by the above dynamic inference algorithm, given the measurement sequences from three positioning systems. Comparing with those in the previous case where only one sequence of probabilistic measurements is available (shown in Figure 4.9(c)), we can see that the measurements from the additional two systems can improve the state estimation significantly, by putting more probability mass to the locations near the ground truth. Similarly, the estimated state $\hat{x}_t$ is superior to that in the static case (shown in Figure 4.6(b)), because the previous and future measurements can influence the estimation of current state through the model dynamics.

4.2 Inference-based Approach for Continuous State Space

Now we consider the case where the state space is continuous. Without loss of generality, we assume that in this case the sensor measurements are also continuous. Throughout this section, we use the environmental monitoring scenario discussed in Section 1.3.2 as the running example, and assume that multiple coexisting sensor systems are monitoring the light intensity of the environment. We assume that the sensor measurements are *probabilistic*, and for simplicity, at a given timestamp t, the sensor measurement reported by the m-th sensor system z_t^m is Gaussian $z_t^m \sim \mathcal{N}(\mu_{z_t^m}, \Sigma_{z_t^m})$. In the example of light intensity

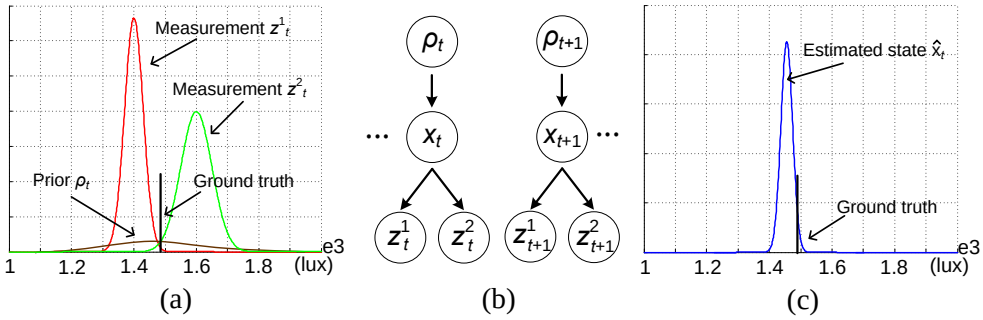


Figure 4.12: (a) The probabilistic sensor measurements z_t^1 and z_t^2 on light intensity reported by two sensor systems sn_1 and sn_2 at time t. Note that here the distribution of the prior ρ_t here is very "flat", indicating that we do not possess strong prior knowledge on the state. (b) The model with probabilistic measurements and prior state distributions used in static inference. (c) The estimated state $\hat{x}_t$ computed by static inference on this model.

which is one dimensional, the measurement z_t^m is reduced to 1D Gaussian over $\mathbb{R}$: however in the following text, we keep the vector notations $\mathcal{N}(\mu_{z_t^m}, \Sigma_{z_t^m})$ to represent the general case over $\mathbb{R}^n$. As in the discrete case, we assume that at some timestamps t, prior state distributions ρ_t are available, which are also assumed to be Gaussian $\rho_t \sim \mathcal{N}(\mu_{\rho_t}, \Sigma_{\rho_t})$. Now we explain the proposed static and dynamic inference techniques for continuous state space in more detail.

Static inference: In the continuous case, static inference also assumes the latent states at different timestamps are independent, and only accesses the measurements and prior state distributions observed at a single timestamp t. Figure 4.12(a) shows two light intensity measurements z_t^1 and z_t^2 reported by systems sn_1 and sn_2, which are probability distributions over $\mathbb{R}$. Note that the existing models such as linear dynamical systems do not explicitly accept probability distributions as observations, but assume the measurements are deterministic values corrupted by certain noise. Typically in those models, an observation z_t^m at time t is a deterministic value, which is a *linear transformation* of the state x_t with additive white noise: $z_t^m = H_t^m x_t + v_t^m$. H_t^m is a matrix that maps the state space to the measurement space, and $v_t^m \sim \mathcal{N}(0, \Sigma_{v_t^m})$ is the noise controlled by the model. Both H_t^m and v_t^m are considered as the model parameters.

In our approach, we also assume that the correlations between measurements and states are linear, but we augment the model by assuming the observation z_t^m to be probabilistic. This can be achieved by setting the covariance of v_t^m to the covariance $\Sigma_{z_t^m}$ of the observed distribution, i.e. in our case $v_t^m \sim \mathcal{N}(0, \Sigma_{z_t^m})$ is no longer a model parameter, but represents the uncertainty in the probabilistic sensor measurement z_t^m. We also assume that the measurements $z_t^{1:M}$ reported by the M coexisting sensor systems at a timestamp t

are conditionally independent given the state x_t, since no external noise or interference is considered. The prior knowledge on states is incorporated in the same way as in the discrete case, where at time t, the prior ρ_t represents the prior belief on the distribution of the state x_t before any measurement is observed: Figure 4.12(a) shows an example of a prior ρ_t, whose distribution is very "flat" since here we do not possess strong prior knowledge on the state. Therefore in this case, a measurement observed at time t only depends on the state x_t, while x_t is influenced by the prior ρ_t, as shown in Figure 4.12(b): the dependencies between the variables are indicated by the directed edges between the nodes.

Based on this augmented model, given the observed measurements $z_t^{1:M}$ and the prior ρ_t, the distribution of the estimated state $\hat{x}_t$ is:

$$p(\hat{x}_t) = p(x_t|z_t^{1:M}, \rho_t) \propto p(x_t, \rho_t) \prod_{m=1}^{M} p(z_t^m|x_t, \rho_t) \tag{4.23}$$

Since all the variables are assumed to be Gaussian, the estimated state $\hat{x}_t$ is naturally Gaussian. Therefore, the inference process can be performed fully on the Gaussian parameters, and in that sense inference for the continuous case is actually *easier* than that in the discrete case. In Equation (A.1), under the linear assumption, $p(z_t^m|x_t, \rho_t)$ is in fact a Gaussian $\mathcal{N}(H_t^m \mu_{x_t}, \Sigma_{z_t^m})$ (note that z_t^m is independent of ρ_t given x_t), and the term $p(x_t, \rho_t)$ is the state distribution in the presence of prior ρ_t, which is also Gaussian. In practice, if ρ_t is available at time t, we assume that $p(x_t, \rho_t) = p(\rho_t)$, otherwise $p(x_t)$ is assumed to uniform.

In our case, the estimated state $\hat{x}_t$ can be computed by modifying the standard Kalman smoother as follows a) the variable v_t^m takes the covariance $\Sigma_{z_t^m}$ of the observed measurement, and b) the distribution of the prior ρ_t is multiplied by the estimated state before any measurement is considered. The detailed derivation is included in Appendix A.1. Figure 4.12(c) shows the distribution of the estimated state computed by this static inference algorithm. We can see that as in the discrete case, fusing measurements from multiple systems can improve state estimation: the estimated state is closer to the ground truth than either of the reported measurements.

Dynamic inference: Unlike static inference, dynamic inference assumes the monitored states are temporally correlated, and estimates the states with all the observed measurements and priors. For simplicity, we assume the correlations between the states at different timestamps are linear: $x_t = F_t x_{t-1} + w_t$, where F_t is the transition matrix and w_t is the process noise, $w_t \sim \mathcal{N}(0, \Sigma_{w_t})$. Both F_t and w_t are assumed to be model parameters known a priori. F_t controls the transition from previous state x_{t-1} to the current state, and w_t determines the uncertainty associated with this state transition.

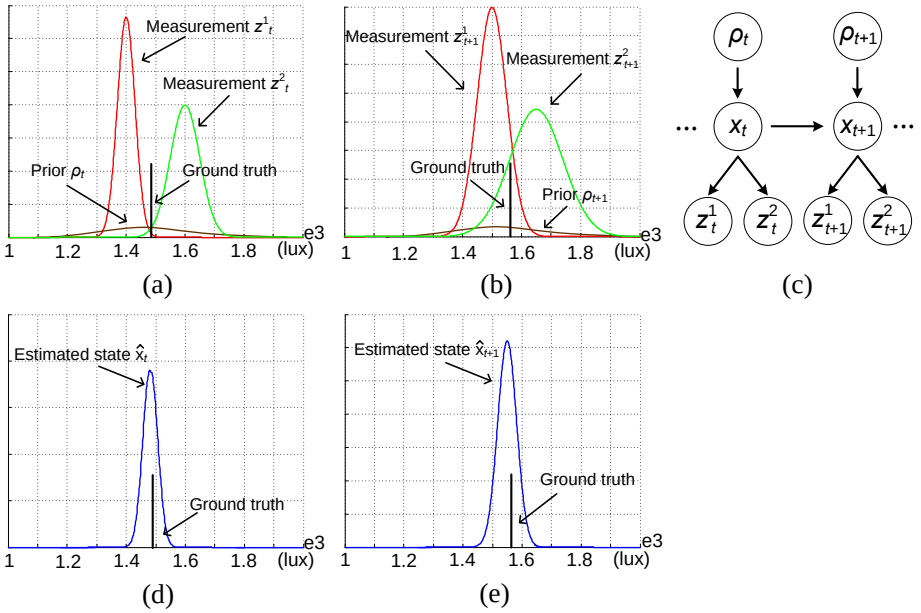


Figure 4.13: (a) – (b) The probabilistic sensor measurements on light intensity reported by two sensor systems sn_1 and sn_2 at time t and $t+1$. (b) The model with probabilistic measurements and prior state distributions used in dynamic inference, where the correlations between states are indicated by the directed edges. (d) – (e) The estimated states at time t and $t+1$ computed by dynamic inference on this model. Note that the estimated state $\hat{x}_t$ is better than that in the static case, since previous and future measurements are taken into account.

As in the static case, we also assume that a measurement z_t^m observed at timestamp t linearly depends on the state x_t: $z_t^m = H_t^m x_t + v_t^m$. But unlike the existing linear dynamical systems which consider v_t^m as a model parameter, in our model v_t^m represents the uncertainty in the observed measurement z_t^m, $v_t^m \sim \mathcal{N}(0, \Sigma_{z_t^m})$. Figures 4.13(a) and (b) show the light intensity measurements reported by two sensor systems sn_1 and sn_2 at timestamps t and $t+1$ respectively, where all the measurements are 1D Gaussian. The priors ρ_t and ρ_{t+1} are also available, as shown in Figures 4.13(a) and (b). Figure 4.13(c) shows the proposed model used in this dynamic case, where the future state x_{t+1} depends on the current state x_t (indicated by the directed edge between x_t and x_{t+1}), and both states are influenced by the corresponding priors ρ_t and ρ_{t+1}.

As in the static case, our dynamic inference algorithm extends the standard Kalman smoother, and uses the variables v_t^m to carry the uncertainty in the measurements rather than using them as fixed model parameters. Also at each timestamp t, the distribution of the prior ρ_t is incorporated before any measurement to bias the estimated state. We leave the detailed derivation in Appendix A.2. Figure 4.13(d) and (e) show the estimated states computed by the proposed dynamic inference algorithm at time t and $t+1$. Comparing

the estimated $\hat{x}_t$ with that in the static case (shown in Figure 4.12(c)), we can see that the $\hat{x}_t$ estimated by dynamic inference is closer to the ground truth, since measurements from previous and future timestamps are taken into account to improve state estimates.

4.3 Impact of State Estimation on Accuracy Estimation

In this section, we discuss how state estimation may influence the quality of accuracy estimation. By definition, an accuracy metric f_ϵ is an error function that maps a probabilistic measurement z_t^m to its accuracy, with respect to the estimated state $\hat{x}_t$. In other words, f_ϵ takes two distributions as input, and produces a scalar value, indicating the *distance* or *difference* between them. Then a natural question arises: if we have an estimated state $\hat{x}_t$ which is close enough to the ground truth x_t (assuming x_t is known) under this function f_ϵ, is the approximated accuracy $f_\epsilon(z_t; \hat{x}_t)$ with respect to $\hat{x}_t$ also close to the real accuracy $f_\epsilon(z_t; x_t)$ with respect to x_t?

Let us first consider the proximity-based accuracy f_ϵ^p, which is defined as the expected Euclidean distance between the measurement and the estimated state: $f_\epsilon^p(z_t; \hat{x}_t) = E[\|z_t - \hat{x}_t\|]$ in Section 3.4.1. Given a probabilistic sensor measurement z_t, an estimated state $\hat{x}_t$ and the ground truth x_t (x_t can be viewed as a random variable with point distribution here), we consider the following two propositions:

Proposition 1. *There exists a positive constant c so that:*

$$|f_\epsilon^p(z_t, x_t) - f_\epsilon^p(z_t, \hat{x}_t)| \leq c f_\epsilon^p(\hat{x}_t, x_t) \tag{4.24}$$

Proposition 2. *For two estimated states $\hat{x}_t$ and $\hat{x}'_t$, if $f_\epsilon^p(\hat{x}_t, x_t) \leq f_\epsilon^p(\hat{x}'_t, x_t)$, then:*

$$|f_\epsilon^p(z_t; \hat{x}_t) - f_\epsilon^p(z_t; x_t)| \leq |f_\epsilon^p(z_t; \hat{x}'_t) - f_\epsilon^p(z_t; x_t)| \tag{4.25}$$

Proposition 1 indicates that the difference between the accuracy approximated by $\hat{x}_t$ and x_t can be *bounded* by the distance between $\hat{x}_t$ and x_t under the metric f_ϵ^p. This means if the estimated state $\hat{x}_t$ is close enough to the ground truth x_t, the estimated accuracy with respect to this $\hat{x}_t$ will not be too different from the real accuracy. We can prove this by exploiting the properties of Euclidean distance.

Proof. We only show the discrete case; the proof for the continuous case is similar. Expanding the left hand side of Equation (4.24) and assuming it is positive without loss of generality, we have:

$$|f_\epsilon^p(z_t; \hat{x}_t) - f_\epsilon^p(z_t; x_t)| = \sum_{x_t} \sum_{z_t} p(z_t, x_t)\|z_t - x_t\| - \sum_{\hat{x}_t} \sum_{z_t} p(z_t, \hat{x}_t)\|z_t - \hat{x}_t\| \tag{4.26}$$

Rearranging the order of summations, we have:

$$|f_\epsilon^p(z_t;\hat{x}_t) - f_\epsilon^p(z_t;x_t)| = \sum_{x_t}\sum_{\hat{x}_t}\sum_{z_t}[p(z_t,\hat{x}_t,x_t)\|z_t - x_t\| - p(z_t,\hat{x}_t,x_t)\|z_t - \hat{x}_t\|] \quad (4.27)$$

Note that for any fixed z_t, $\hat{x}_t$ and x_t, the triangle inequality $\|z_t - x_t\| - \|z_t - \hat{x}_t\| \leq \|\hat{x}_t - x_t\|$ holds for the Euclidean distance. Substituting this into Equation (4.27) and summing out the variable z_t, we have:

$$\begin{aligned} |f_\epsilon^p(z_t;\hat{x}_t) - f_\epsilon^p(z_t;x_t)| &\leq \sum_{x_t}\sum_{\hat{x}_t}\sum_{z_t}[p(z_t,\hat{x}_t,x_t)\|\hat{x}_t - x_t\|] \\ &= \sum_{x_t}\sum_{\hat{x}_t}[p(\hat{x}_t,x_t)\|\hat{x}_t - x_t\|] \end{aligned} \quad (4.28)$$

which is exactly the expansion of the right hand side of Eqn. (4.24) in the case of $c = 1$. □

Proposition 2 is actually stronger than Proposition 1. It requires that the difference between the estimated accuracy $f_\epsilon^p(z_t,\hat{x}_t)$ and real accuracy $f_\epsilon^p(z_t,x_t)$ can be monotonically determined by the expected Euclidean distance between the estimated state $\hat{x}_t$ and x_t. Unfortunately, this is not always true, and we can prove this by finding the following simple counterexample, even for the deterministic measurements:

Proof. Let the sample space Ω be the 2D Euclidean space. Consider the special case where the measurement z_t, the ground truth x_t, and two different state estimates $\hat{x}_t$ and $\hat{x}'_t$ are points on the 2D plane, where they have the following coordinates: $z_t = (5,1)$, $x_t = (1,1)$, $\hat{x}_t = (2,1)$, and $\hat{x}'_t = (1,3)$, as shown in Figure 4.14. Then in this case, we have: $f_\epsilon^p(\hat{x}_t;x_t) = 1 \leq f_\epsilon^p(\hat{x}'_t;x_t) = 2$, but: $|f_\epsilon^p(z_t;\hat{x}_t) - f_\epsilon^p(z_t;x_t)| = 1 \geq |f_\epsilon^p(z_t;\hat{x}'_t) - f_\epsilon^p(z_t;x_t)| = \sqrt{20} - 4$. □

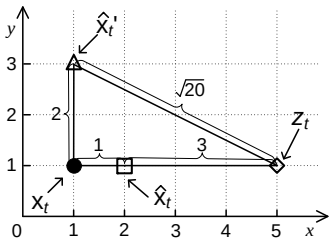


Figure 4.14: A counterexample to Proposition 2

Thus for the proximity-based accuracy metric, it is possible to have an estimated state $\hat{x}_t$ which is closer to the ground truth x_t than another estimate $\hat{x}'_t$, but the accuracy estimated with respect to $\hat{x}_t$ is worse than that of $\hat{x}'_t$. Unfortunately, for the similarity-based accuracy metric f_ϵ^s neither of the two properties holds, since KL divergence itself is not a *true metric*: e.g. it is not symmetric and does not satisfy the triangle inequality. Therefore, there is no guarantee that the difference between the accuracy estimated with respect to $\hat{x}_t$ and x_t, i.e. $|f_\epsilon^s(z_t;\hat{x}_t) - f_\epsilon^s(z_t;x_t)|$, can be monotonically determined, or bounded by the distance between the state estimate $\hat{x}_t$ and ground truth x_t.

4.4 Discussion

In this chapter, we have proposed a novel inference-based accuracy estimation approach, which implements the general accuracy estimation framework discussed in the previous chapter. We have shown that by inferring the latent states, one can reason about the accuracy of sensor systems accordingly. Unlike the existing work, we have considered a new setting, where multiple coexisting sensor systems report probabilistic measurements, and at certain timestamps we may possess prior knowledge on the states. Existing models and inference algorithms fail to work in this context, because they only accept deterministic observations, and do not consider prior knowledge on states other than at the beginning.

We have proposed augmented models for both discrete and continuous cases, to capture the new elements in our context. The proposed models can accept multiple probabilistic sensor measurements, take prior knowledge on states at arbitrary timestamps, and have parameters with new semantics. Based on the proposed models, we have designed novel inference algorithms, which are able to estimate the latent states with multiple sequences of probabilistic observations and priors, in both discrete and continuous cases. Concretely, the proposed inference algorithms process the probabilistic measurements by marginalisation (in discrete case) or redefining the model parameters (in continuous case), and for multiple probabilistic measurements at a timestamp, the proposed algorithms incorporate them by assuming conditional independence between measurements from different systems. In addition, the proposed inference algorithms can bias the estimated states by taking the priors into account before any measurements are considered.

We have also demonstrated two different variants of the proposed inference approach, the static inference and dynamic inference, depending on whether we exploit the temporal correlations between the states. The proposed static inference algorithms estimate states with measurements and priors within single timestamps and ignore any correlations between adjacent states. On the other hand, the dynamic inference algorithms consider the states as a stochastic process, and estimate the states with the full sequences of observed measurements and priors. We have discussed the complexity of the proposed algorithms in all cases, and shown that they are not prohibitively expensive compared to existing ones.

We have shown that for the inference task, a) dynamic algorithms are generally better than the static counterparts since they consider the data observed at different timestamps; b) incorporating probabilistic measurements is beneficial since they typically carry more information than the deterministic ones; and c) measurements from multiple sensor systems can validate each other, and thus improve state estimation significantly. Chapter 6

will confirm the above findings with extensive experiments performed on two real sensor datasets. Finally, we have illustrated that the quality of the estimated states can have substantial impact on the quality of accuracy estimation.

However, the inference-based approach proposed in this chapter estimates the states with observed sensor measurements and priors under certain model parameters, which are assumed to be known a priori. In discrete models the parameters are the transition probabilities a_{ij} and emission probabilities $b_j^m(k)$, while in continous models the parameters are the transition matrices F_t, covariance of the process noise Σ_{w_t}, and the measurement model H_t^m. As we can see from the derivations of the proposed approach in different cases, those parameters actually play a vital role when estimating the latent states. Therefore, the performance of the proposed inference-based approach may be limited by the quality of model parameters (Chapter 6 will show experimental results that confirm this). In the next chapter, we will propose a learning-based accuracy estimation approach, which is able to learn the appropriate model parameters from the observed data, and thus improve accuracy estimation.

Chapter 5

Learning-based Approach for Accuracy Estimation

In this chapter, we present the learning-based approach for accuracy estimation. The learning-based approach also considers the monitored states, the sensor measurements and the prior knowledge on states within a probabilistic model. However, unlike the inference-based approach discussed in the previous chapter, the learning-based approach does not rely on any model parameters known a priori, but uses learning techniques to address the accuracy estimation problem. We consider the same setting as discussed in Chapter 4, where multiple sensor systems sn_1, ..., sn_M are monitoring the underlying states. The states x_t are random variables defined on sample space Ω, and indexed with discrete timestamps $t = 1 : T$. At a given timestamp t, the m-th sensor system reports a probabilistic measurement z_t^m, which is also a random variable with observed probability distribution $p(z_t^m)$ defined on Ω, e.g. an estimated position with error ellipse, or an estimated temperature value with confidence interval. We assume that at some timestamps t, we may possess prior knowledge on how the state x_t is distributed. For instance, personal calendars may reveal the likely positions of a user during a certain period, e.g. joining a group meeting or attending a talk. In this thesis, such information is denoted as a random variable ρ_t, whose distribution $p(\rho_t)$ can carry such information.

The proposed learning-based accuracy estimation approach has two different variants. The first variant is designed for the case where the sensing application aims to index the accuracy of sensor systems over attributes other than the monitored states. In this case, the learning-based approach follows the four-layer accuracy estimation framework introduced in Chapter 3. However different from the inference-based approach, in the state estimation layer, the proposed learning-based approach firstly learns the model parameters that are the most consistent with the observed sensor measurements and priors, and then

uses the learned parameters to estimate states. The accuracy of the sensor measurements is evaluated and then aggregated as discussed in Chapter 3.

On the other hand, the second variant of the proposed learning approach is designed for the case where the sensing application seeks to assess the accuracy of the sensor systems indexed over the monitored states, which cannot be addressed by the inference-based approach. In this case, the proposed learning approach jointly solves the problems of accuracy estimation and indexing, and casts them into that of a parameter estimation problem. The problem is then solved by the proposed parameter learning algorithms based on the Expectation Maximisation (EM) scheme (we have briefly explained the EM scheme in Chapter 3, and will provide more details later in this chapter). Concretely, the technical contributions of this chapter are as follows:

- We propose a novel learning-based approach for accuracy estimation. The proposed approach is able to implement the general accuracy estimation framework, and by employing learning techniques, it improves the estimation of accuracy compared to the inference-based approach. We also show that in the case where accuracy is required to be indexed over the monitored states, where the inference approach fails to work, the proposed learning approach can estimate this accuracy by jointly addressing the problems of accuracy estimation and indexing.

- We propose new parameter learning algorithms for both variants of the learning-based approach. The proposed algorithms can incorporate multiple sequences of probabilistic measurements, and take prior knowledge on arbitrary states into account. We show two different types of parameter learning algorithms, the static learning and dynamic learning, depending on whether the temporal correlations between the states are considered.

- We discuss several important implementation details of the proposed learning-based approach. We show that the initial model parameters can be estimated empirically with the observed data by trusting the reported sensor measurements. We also show how to address the common problem of insufficient training data and model overfitting.

This chapter is organised as follows. Section 5.1 provides an overview of the proposed learning-based accuracy estimation approach, and explains the two variants of the approach in more detail. We show that for both variants, the core task is to learn the parameters of the model. Then Section 5.2 discusses the problem of parameter learning for the discrete case, which includes both the static and dynamic parameter learning algorithms designed for

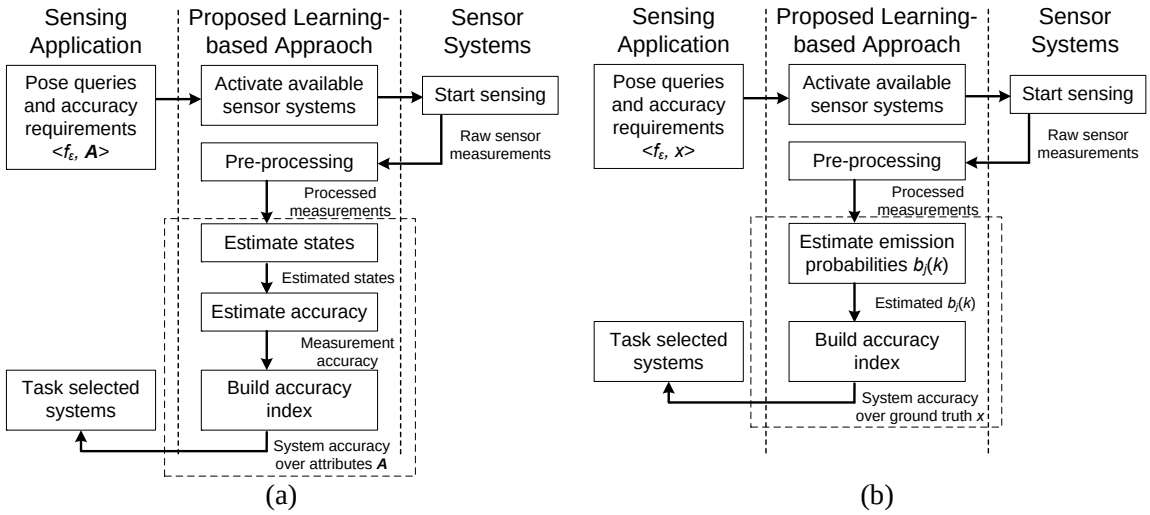


Figure 5.1: The workflow of the two variants of the proposed learning-based accuracy estimation approach, where the highlighted parts show their differences. (a) The variant in which the accuracy is indexed over attributes $\boldsymbol{A}$ other than the monitored states. This variant implements the four-layer accuracy estimation framework. (b) The variant in which the accuracy is indexed over the monitored states. The state estimation layer is skipped, and accuracy is estimated directly from the learned model parameters.

our context. Section 5.4 explains how to modify the existing learning algorithms, e.g. the Kalman learner for continuous models, The detailed derivation is included in Appendix B. Section 5.4 discusses several important implementation issues of the proposed learning-based approach, and Section 5.5 concludes this chapter.

5.1 Overview of the Learning-based Approach

As stated above, the proposed learning-based accuracy estimation approach has two different variants, designed for the cases where a) the sensing application seeks to index the accuracy of sensor measurements over attributes other than the monitored states, and b) the sensing application would like to index the accuracy of sensor measurements over the monitored states. Subsections 5.1.1 and 5.1.2 will explain the two variants in more detail.

5.1.1 Accuracy indexed over non-state attributes

We first consider the case that the sensing application requires the accuracy of sensor measurements to be indexed over non-state attributes. As discussed in Section 3.5, an attribute is a property attached to a sensor measurement, such as the timestamp when it is reported,

the location where it is observed, etc. This thesis assumes that the attributes are discrete and with finite domains. The event that an attribute A is assigned to a particular value in its domain $a \in dom_A$ provides information on the *context* in which the sensor measurement is made. We have shown in Section 3.5 that such an attribute assignment $A = a$ can be used to *group* multiple sensor measurements, and the accuracy of a sensor system under assignment $A = a$ is evaluated as the *average accuracy* of the measurements grouped by $A = a$.

In this case, the problem of accuracy estimation can be addressed by the inference-based accuracy estimation approach, as shown in Chapter 4. However, as we discussed at the end of the previous chapter, the inference-based approach relies heavily on the model parameters known a prior, and may perform badly when the parameters are poor (we will show this with experiments on real sensor datasets in Chapter 6). Therefore, the proposed learning-based approach addresses this limitation by incorporating learning techniques to find the appropriate model parameters, and thus improving the estimation of accuracy.

Concretely in this case, the proposed learning-based approach implements the four-layer accuracy estimation framework, where the reported sensor measurements are firstly cleaned through the pre-processing layer. In the state estimation layer, the proposed learning-based approach does not rely on the known model parameters, but firstly tries to learn the parameters that are the most consistent with the reported sensor measurements and observed priors. It then uses the learned parameters to estimate the latent states as in the inference-based approach. With the estimated states, the accuracy of the measurements is computed by the accuracy estimation layer, and further aggregated over the given attributes to build the accuracy indices in the accuracy indexing layer, as discussed in Chapter 3. Figure 5.1(a) shows the workflow of the proposed learning-based approach for this case.

5.1.2 Accuracy index over monitored states

The second variant of the proposed learning-based approach is designed for the case where sensing application seeks to index accuracy over the monitored states (for simplicity, this thesis only allows the accuracy to be indexed over discrete states). This is common in practice: consider the example of indoor positioning, where multiple positioning systems track a given user. The accuracy of the positioning systems can vary significantly over different locations, e.g. a system may be very accurate in the atrium, while another system can perform well at the corridors. In this case, the sensing application may want to know how the accuracy of a positioning system varies over space, and with this knowledge it can make informed decisions as to which system to task when the user moves across the environment. Therefore the goal here is to build accuracy indices of the positioning systems

over the actual locations of the user, which are the monitored states. However, this is not straightforward since the real states (the actual trajectories of the user) are not known exactly when the measurements are made. The inference-based approach cannot address this, because although the accuracy of sensor measurements can be evaluated with respect to the estimated states, it is still impossible to build the accuracy indices over the *unknown* ground truth.

One possible approach is to use the estimated states to build the accuracy indices. For instance, one can always use the means of the estimated states $\hat{x}_t$ as if they were the ground truth, and aggregate the evaluated accuracy over them. This approach will introduce new errors, since the means may not accurately represent real state. A better approach is to combine the error-in-variable regression techniques (e.g. the methods surveyed in [138, 139]) with the inference-based approach, in order to incorporate the uncertainty in the estimated state $\hat{x}_t$ when indexing accuracy. However, the drawback is that the accuracy metrics (e.g. the proximity-based accuracy metric proposed in this thesis) can be highly non-linear, and thus this approach may not have a closed form solution, but needs to use computationally expensive approximation techniques. The proposed learning-based approach avoids those limitations, and considers a joint problem of accuracy estimation and indexing, which is then solved by learning the model parameters. Figure 5.1(b) shows the workflow of the proposed learning-based approach for this case, where the step of state estimation is skipped, and the accuracy indices are derived from the learned model parameters directly. Now we explain why the model parameters are key to estimating accuracy.

Let us consider the indoor positioning example, and assume that the sensing application seeks to index the accuracy of positioning systems over discrete locations. At a time t, the m-th positioning system generates a probabilistic measurement z_t^m: $[z_t^m(1), ..., z_t^m(N)]$, where $z_t^m(k)$ indicates the belief of the system that the user is at location l_k, $1 \leq m \leq M$, $1 \leq k \leq N$. Recall from the previous chapter, that the expected probability a location l_k is reported in measurements given the state is l_j is defined as the *emission probability* $b_j^m(k)$, and is regarded as a model parameter. Now we explain under the two different accuracy metrics, the proximity-based and the similarity-based, how the emission probabilities $b_j^m(k)$ are related to the accuracy of sensor systems.

Proximity-based accuracy metric: The proximity-based accuracy $f_\epsilon^p(z_t^m; \hat{x}_t)$ of a measurement z_t^m is defined as the expected Euclidean distance between the measurement and estimated state $\hat{x}_t$. Let us first assume that the state x_t, i.e. the actual position of the user at time t is known: $x_t = l_j$, $1 \leq j \leq N$. Then given z_t^m, its real accuracy $f_\epsilon^p(z_t^m; x_t)$ can be computed as follows (note that x_t is a known single location l_j while z_t^m is a distribution of

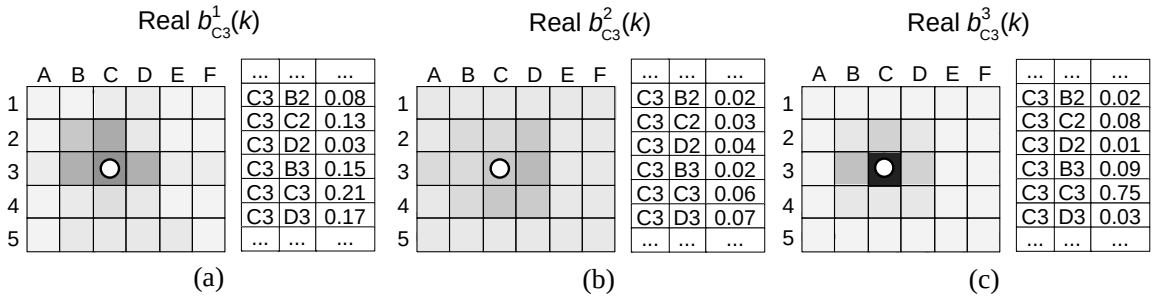


Figure 5.2: The real emission probabilities $b^m_{\text{C3}}(k)$ for location C3 (computed with the ground truth) of three sensor systems sn_1 (a), sn_2 (b) and sn_3 (c). Grey blocks indicate the value of $b^m_{\text{C3}}(k)$ for different locations l_k, where darker blocks indicate more probability mass. $b^m_{\text{C3}}(k)$ are also shown in the conditional probability tables on the right.

the N locations):

$$f^p_\epsilon(z^m_t; x_t) = E[\|z^m_t - x_t\|] = \sum_{k=1}^{N} z^m_t(k)\|l_k - l_j\| \tag{5.1}$$

where $\|l_k - l_j\|$ is the Euclidean distance between the location l_k in the measurement z^m_t and location $x_t = l_j$, while $z^m_t(k)$ is the probability assigned to l_k by z^m_t. Given that the ground truth x_t is known in this case, the accuracy of sensor system sn_m at location $x_t = l_j$ is the average accuracy of all measurements z^m_t with ground truth equal to l_j (see the definition in Section 3.5). Let us denote the accuracy of system sn_m at location l_j as $I^m_\epsilon(j)$. It is evaluated as follows:

$$I^m_\epsilon(j) = \frac{1}{c} \sum_{\substack{t=1 \\ s.t. x_t = l_j}}^{T} \Big[\sum_{k=1}^{N} z^m_t(k)\|l_k - l_j\| \Big] \tag{5.2}$$

where c is a normalising constant, i.e. the number of timestamps that $x_t = l_j$. Changing the order of the summations, $I^m_\epsilon(j)$ can be represented as:

$$I^m_\epsilon(j) = \sum_{k=1}^{N} \Big[\frac{1}{c} \sum_{\substack{t=1 \\ s.t. x_t = l_j}}^{T} z^m_t(k) \Big] \|l_k - l_j\| \tag{5.3}$$

where for a given k, the inner summation is in fact the average of the k-th probability of all $z^m_t(k)$ when the ground truth x_t is l_j. As the time period T increases, this probability can be viewed as the expected probability of reporting location l_k in the measurements given that the real position is at l_j, which is exactly the emission probability $b^m_j(k)$ defined in our augmented model in Chapter 4. Therefore in the case when the ground truth x_t is

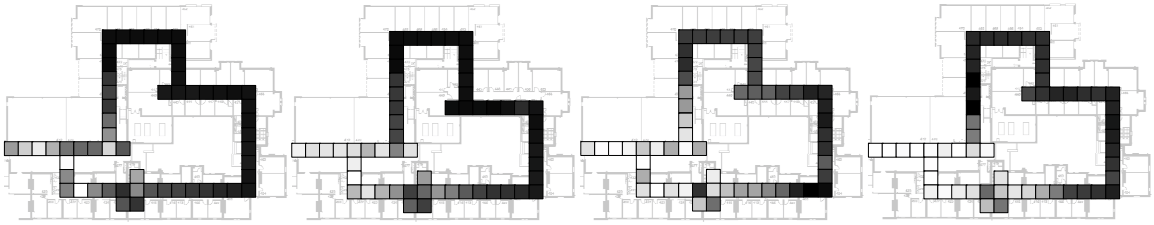
unknown, the accuracy of system sn_m at location l_j can be approximated with the emission probabilities as follows:

$$\hat{I}_{\epsilon}^{m}(j) = \sum_{k=1}^{N} b_j^m(k) \|l_k - l_j\| \tag{5.4}$$

Note that given the map of the indoor environment, $\|l_k - l_j\|$ is constant for any j, k, and thus the accuracy of the system at location l_j is fully determined by the emission probabilities $b_j^m(k)$. In fact, $\hat{I}_{\epsilon}^{m}(j)$ can be viewed as a weighted sum of the distances between the reported location l_k and the ground truth l_j: the more weights assigned to locations that are close to the ground truth, the more accurate the system is. Figure 5.2 shows the real emission probabilities of three different systems for location C3 (computed assuming knowledge of the ground truth). As we can see in Figure 5.2(b) when the state is at location C3, the emission probabilities $b_{C3}^2(k)$ are distributed quite uniformly, which means sn_2 has similar probability of reporting different locations when the state is in C3. On the other hand, as shown in Figure 5.2(a), we can see that system sn_1 is more likely to report locations that are close to the ground truth, and according to Equation (5.4), sn_1 should be more accurate than sn_2 at location C3. Figure 5.2(c) shows when the ground truth is C3, system sn_3 has a very large probability (0.7) of reporting this location C3 in its measurements, and thus is the most accurate among the three systems. Therefore, under the proximity-based accuracy metric, the problem of indexing accuracy over the monitored states can be solely addressed by learning the emission probabilities $b_j^m(k)$ of the coexisting systems. Figures 5.3 (a1) and (a2) confirm this: Figure 5.3 (a1) shows the real proximity-based accuracy of a positioning system at different locations (computed with the ground truth), while Figure 5.3 (a2) shows the accuracy derived from the learned emission probabilities (computed by the proposed dynamic learning algorithm, which will be explained shortly). We can see that they are very similar, which means the learned emission probabilities can be used to approximate the real accuracy.

Similarity-based accuracy metric: Now we show that under similarity-based accuracy metric f_{ϵ}^{s}, it is also possible to use the emission probabilities $b_j^m(k)$ to approximate the accuracy $I_{\epsilon}^{m}(j)$. The similarity-based accuracy $f_{\epsilon}^{s}(z_t^m; \hat{x}_t)$ of a measurement z_t^m is defined as the KL divergence between $p(z_t^m)$ and the distribution $p(\hat{x}_t)$ of the estimated state. Let us also assume the state x_t is known, and in this case $x_t = l_j$, then the accuracy of a measurement z_t^m is given by:

$$\begin{aligned} f_{\epsilon}^{s}(z_t^m; x_t) &= D_{\mathrm{KL}}(p(x_t)||p(z_t^m)) \\ &= \log(\frac{P(x_t = l_j)}{P(z_t^m = l_j)}) P(x_t = l_j) = -\log(z_t^m(j)) \end{aligned} \tag{5.5}$$



(a1) (a2) (b1) (b2)

Figure 5.3: The accuracy of a positioning system indexed over different locations, where a lighter block means more accurate. (a1) The real proximity-based accuracy evaluated with respect to the ground truth. (a2) The approximated proximity-based accuracy evaluated using the learned emission probabilities $b_j^m(k)$. (b1) The real similarity-based accuracy evaluated with respect to the ground truth. (b2) The approximated similarity-based accuracy evaluated using the learned emission probabilities $b_j^m(j)$.

Note that here $P(x_t = l_j) = 1$ since x_t is known to be l_j. Similarly, in this case the accuracy of system sn_m at l_j is given by the average accuracy of all measurements whose ground truth x_t is l_j:

$$I_\epsilon^m(j) = \frac{1}{c} \sum_{\substack{t=1 \\ s.t. x_t = l_j}}^{T} -\log(z_t^m(j)) = -\log[g_{x_t=l_j}(z_t^m(j))] \tag{5.6}$$

where c is the number of timestamps when the state x_t is l_j, and $g_{x_t=l_j}(z_t^m(j))$ is the geometric mean of the measured probabilities $z_t^m(j)$ of those measurements whose ground truth x_t is l_j. In the case where x_t is unknown, we can also approximate the accuracy $I_\epsilon^m(j)$ with the emission probabilities as the time period T increases:

$$\hat{I}_\epsilon^m(j) = -\log(b_j^m(j)) \tag{5.7}$$

which only depends on the elements $b_j^m(j)$ of the emission probabilities. Therefore, under the similarity-based accuracy metric, the problem of estimating the accuracy of sensor system sn_m indexed over monitored states can be also cast into that of learning the emission probabilities. Figure 5.3(b1) shows the real similarity-based accuracy evaluated with the ground truth, while Figure 5.3(b2) shows the accuracy approximated by learning $b_j^m(j)$ as discussed above. We also observe that Figure 5.3(b2) resembles Figure 5.3(b1), which indicates the real similarity-based accuracy can be effectively approximated by the learned emission probabilities.

To sum up, in both cases, where the accuracy of sensor systems is required to be indexed over the non-state attributes (as discussed in Section 5.1.1), or over the monitored states (as discussed in Section 5.1.2), the key task of the proposed learning-based accuracy estimation

approach is to find the appropriate model parameters. In Sections 5.2 and 5.3, we will explain the proposed parameter learning techniques for the discrete and continuous cases respectively.

5.2 Parameter Learning for Discrete State Space

We first consider the task of parameter learning in the case where the state space is discrete. We assume the sample space Ω of both the monitored states and the sensor measurements is a finite discrete set. Throughout this section we consider the indoor positioning scenario as the running example, and of course the proposed techniques can be used in many other application scenarios. We use the identical settings as in Section 4.1, where multiple indoor positioning systems provide measurements on the locations of a user. The indoor environment is represented by a finite set L of N locations, e.g. different rooms or corridor segments.

As discussed in Section 3.3.3, the parameter learning problem of the general latent variable models can be addressed by the Expectation Maximisation (EM) scheme. Let X be the set of latent variables, i.e. the states in our case, and O be the set of observed data, i.e. the sensor measurements and priors. The goal of the EM scheme is to find the maximum likelihood estimate (MLE) of the model parameters θ, which maximise the likelihood of the observed data O. To achieve this, the first step is to formulate the likelihood function $L(\theta; O, X) = P(O, X|\theta)$, so that the ML estimate of the model parameters $\hat{\theta}_{\text{ML}}$ is given by summing out all latent variables X in the likelihood function L:

$$\hat{\theta}_{\text{ML}} = \arg\max_{\theta} \sum_{X} L(\theta; O, X) \tag{5.8}$$

Given the derived likelihood function, the EM scheme iteratively performs the following two steps to find the maximum likelihood estimate of the parameters (see Figure 3.5 in Section 3.3.3):

- The *E*-step, which derives the expected log likelihood function $Q(\theta', \theta)$ of the observed data. The expectation is taken with respect to the conditional distribution of the states X given the observed data O, under the current parameters θ:

$$Q(\theta', \theta) = E_{X|O,\theta}[\log L(\theta'; O, X)] \tag{5.9}$$

- The *M*-step, which finds the new parameters θ' that maximise the derived likelihood function of the observed data.

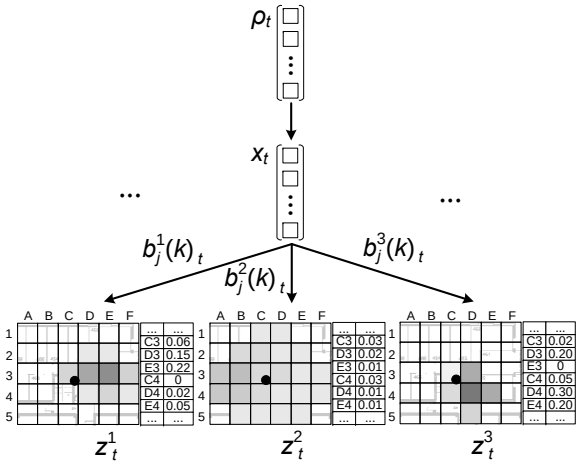


Figure 5.4: The model used in static learning. At time t, the state x_t is influenced by the prior ρ_t, and emits three probabilistic measurements z_t^1, z_t^2 and z_t^3, each of which is a probability distribution over the locations. The model parameters are the emission probabilities $b_j^1(k)_t$, $b_j^2(k)_t$ and $b_j^3(k)_t$ of the three sensor systems.

It can be shown that in each iteration, the parameters θ' that maximise the Q function always increase the marginal likelihood of data: i.e. $\max_{\theta'} Q(\theta', \theta)$ implies $\sum_X L(\theta'; O, X) \geq \sum_X L(\theta; O, X)$. Therefore, the above EM scheme will eventually converge to a local optimum; however, there is no guarantee that it will converge to the global optimum. In practice, the converged values are typically used as the learned parameters. In this chapter, we consider two different parameter learning techniques, the static parameter learning and the dynamic parameter learning, depending on whether we exploit the temporal correlations between the states. The next two subsections will explain the proposed static and dynamic parameter learning techniques in more detail.

5.2.1 Static parameter learning

Static parameter learning ignores any temporal correlations between the monitored states, and operates on single timestamps (slices) of the data. At a given time t, static learning exploits the probabilistic measurements $z_t^{1:M}$ reported by the M sensor systems and the prior ρ_t, and learns the model parameters that are the most consistent with the observed data at time t. Figure 5.4 shows the model used in static learning. Unlike the standard Bayesian networks, at time t the state x_t in our model emits multiple probabilistic measurements, and is also influenced by the prior ρ_t (as indicated by the directed edge between ρ_t and x_t). We now show how to formulate and address the static parameter learning problem based on the Expectation Maximisation (EM) framework discussed above.

Observed data: Unlike the existing learning techniques which re-estimate the model parameters only based on deterministic observations, the observed data O_t considered in our case includes multiple probabilistic measurements $z_t^{1:M}$ and priors ρ_t: $O_t = \{z_t^{1:M}, \rho_t\}$. In the positioning context, a measurement z_t^m is a vector $[z_t^m(1), ..., z_t^m(N)]$, where $z_t^m(k)$ is the belief of the system that the user is in location l_k. The prior ρ_t represents the prior knowledge about the monitored state x_t, which is also represented by a vector of probabilities $[\rho_t(1), ..., \rho_t(N)]$. The term $\rho_t(j)$ is the prior belief that the user should be in location l_j at time t.

Model parameters: In this case, the model parameters θ that need to be learned are the emission probabilities $b_j^m(k)_t$ at each timestamp t, as shown in Figure 5.4. In our model, the $b_j^m(k)_t$ is the *expected probability* of having l_k in the measurements from system sn_m given that the state is l_j at time t. Note that the $b_j^m(k)_t$ here has a different meaning from that of the standard Bayesian networks, which typically consider deterministic observations (as discussed in Section 4.1.1.1). In those models, the emission probabilities represent the likelihood of making an observation given the state, and therefore the probability of observing a measurement given the state can be fully determined by the emission probabilities. However in our case, such probability cannot be solely captured by the emission probabilities, since now the measurement z_t^m is a vector, and $P(z_t^m|x_t)$ cannot be modeled by any finite distribution.

Likelihood function: Given the observed data O_t and the state x_t, the likelihood function is formulated as:

$$\begin{aligned} L(\theta; O_t, x_t) &= P(z_t^{1:M}, \rho_t, x_t|\theta) \\ &= P(\rho_t, x_t) \prod_{m=1}^{M} P(z_t^m|\rho_t, x_t, \theta) \end{aligned} \tag{5.10}$$

under the assumption that measurements from different systems are independent conditioned on the state x_t. The term $P(\rho_t, x_t)$ is in fact the prior belief of state x_t before observing any measurement, which is specified by the prior ρ_t. The term $P(z_t^m|\rho_t, x_t, \theta)$ is the probability of observing measurement z_t^m given the state x_t under the current parameters θ (note that z_t^m and ρ_t are independent), and can be evaluated with the observed distribution $p(z_t^m)$ and the current emission probabilities $b_j^m(k)_t$.

***E*-step:** As we explained above, when both the states and the model parameters are unknown, the EM scheme firstly "pretends" to know the parameters and uses them to evaluate the states. The estimated states are then used to derive the expected log likelihood function Q (we will show how the new parameters are computed by maximising the Q function

shortly). Given the likelihood function $L(\theta; O_t, x_t)$, in each learning iteration the E-step evaluates the Q function as follows:

$$\begin{aligned} Q(\theta', \theta) &= E_{x_t|O_t,\theta}[\log L(\theta'; O_t, x_t)] \\ &= \sum_{j=1}^{N} \underbrace{P(x_t = l_j | z_t^{1:M}, \rho_t, \theta)}_{\hat{x}_t(j)} \log[\underbrace{P(z_t^{1:M}, \rho_t, x_t = l_j | \theta')}_{A}] \end{aligned} \tag{5.11}$$

where we sum over all possible locations l_j for the latent state x_t. Recall that in the previous chapter, for a given l_j, the term $P(x_t = l_j | z_t^{1:M}, \rho_t, \theta)$ here is the j-th probability in the distribution of the estimated state $\hat{x}_t$, which is denoted as $\hat{x}_t(j)$. Note that $\hat{x}_t(j)$ is evaluated with the current model parameters θ. On the other hand, the term A is the joint probability of the measurements, prior and state under the new model parameters θ'.

Let us first consider the simple case where the measurements are deterministic (single locations), and no prior is available. Then the term A can be evaluated as the product of the emission probabilities: $\prod_{m=1}^{M} b_j^m(z_t^m)'_t$. Note that here $b_j^m(z_t^m)$ represents the likelihood of observing the measurement z_t^m given the state l_j. However, in our case where the measurements are probabilistic and prior ρ_t is available, the term A becomes:

$$P(z_t^{1:M}, \rho_t, x_t = l_j | \theta') = \rho_t(j) \prod_{m=1}^{M} \sum_{k=1}^{N} z_t^m(k) b_j^m(k)'_t \tag{5.12}$$

where the emission probabilities $b_j^m(k)'_t$ have a different meaning than those in the case of deterministic measurements (as we discussed in static inference): here $b_j^m(k)'_t$ is the expected probability that location l_k is reported in measurements of sn_m given the state is l_j.

***M*-step:** The derived $Q(\theta', \theta)$ is actually a function of the new emission probabilities $b_j^m(k)'_t$, and the estimated state $\hat{x}_t$ computed with the old parameters $b_j^m(k)_t$. Then the goal of this *M*-step is to find the parameters $b_j^m(k)'_t$ that maximise the Q function: $b_j^m(k)'_t = \arg\max_{b_j^m(k)'_t} Q(\theta', \theta)$. In other words, the EM scheme estimates the latent states with the "guessed" model parameters in the above *E*-step, and in the *M*-step, it re-calibrates the model parameters with the estimated states. In this case, taking the stochastic constraint $\sum_{k=1}^{N} b_j^m(k)'_t = 1$ into account, the new parameters $\theta' = \{b_j^m(k)'_t\}$ that maximise the Q function are given by:

$$b_j^m(k)'_t = \frac{s_t^m(j,k)}{\hat{x}_t(j)} \tag{5.13}$$

where the denominator $\hat{x}_t(j)$ is the posterior probability of being in state l_j, estimated based on measurements from all systems and the prior under current model parameters.

The nominator $s_t^m(j,k)$ on the other hand, is the probability that the state is l_j and the system sn_m reports l_k.

In the simple case where the measurements are deterministic (assuming z_t^m is l_k and uniform prior distribution for x_t), $s_t^m(j,k)$ is actually $b_j^m(k)_t \prod_{\tilde{m}=1}^{M} b_j^{\tilde{m}}(z_t^{\tilde{m}})_t$, $\tilde{m} \neq m$. Here $b_j^m(k)_t$ represents the probability that system sn_m reports l_k when the state is l_j, and the product is the marginal probability of having all observations from other systems when state is l_j. However in our case, $s_t^m(j,k)$ is different since the observations are probabilistic, and $s_t^m(j,k)$ is given by:

$$s_t^m(j,k) = \rho_t(j) \underbrace{z_t^m(k) b_j^m(k)_t}_{B} \underbrace{\prod_{\tilde{m}=1}^{M} \sum_{i=1}^{N} z_t^{\tilde{m}}(k) b_j^{\tilde{m}}(i)_t}_{C}, \tilde{m} \neq m \tag{5.14}$$

where $\rho_t(j)$ is the prior. Term B is the probability that system sn_m associates with l_k in the measurement when the state is l_j. $z_t^m(k)$ is the observed probability assigned to l_k, and $b_j^m(k)_t$ here represents the expected probability that l_k is in the measurement given the state is l_j. Term C is the marginal probability of having all observations from other systems when state is l_j. Note that unlike the deterministic case, here we have to enumerate all possible locations l_i.

Intuitively, in each iteration static learning firstly estimates the state with all measurements and prior, and then uses the estimated state and observed measurements to re-estimate the emission probabilities for each system. The new emission probabilities are evaluated based on counting: given the states estimated under the current model parameters, we count the expected probability that a system sn_m reports a location l_k when the state is l_j, and divide it with the total probability of being in state l_j.

Complexity: As explained above, the proposed static learning algorithm is different from the existing EM-based learning algorithm on standard Bayesian networks, in the sense that it considers multiple probabilistic observations at a timestamp, and uses prior knowledge on state to bias the learned model parameters. Therefore, it is more expensive: the complexity of this static learning algorithm is $O(ITMN^2)$ (existing algorithms are typically $O(ITN)$), where I is the number of learning iterations, T is the number of timestamps, M is the number of sensor systems, and N is the number of locations. This is because each learning iteration needs to evaluate the distribution of $\hat{x}_t$, which is in fact performing static inference on the augmented model (as discussed in Section 4.1.1.3), which has $O(TMN^2)$ complexity.

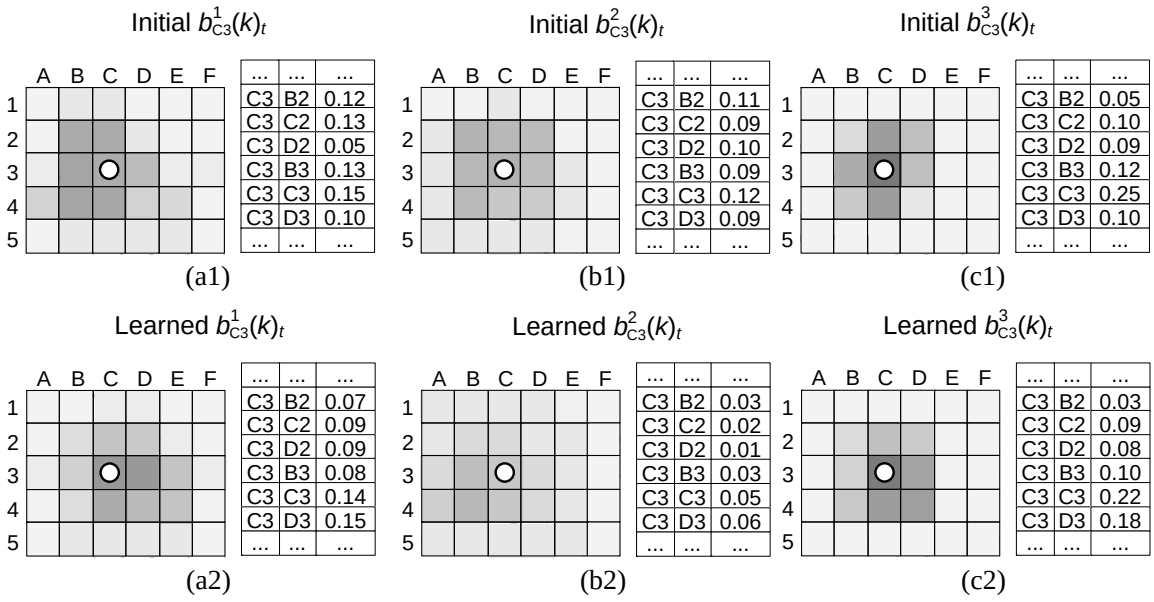


Figure 5.5: The initial emission probabilities at location C3 (on top) and the new emission probabilities learned by static learning (on bottom). The learned emission probabilities are clearly influenced by the measurements z_t^1, z_t^2 and z_t^3 observed at time t (shown in Figure 5.4).

Example: Let us consider a simple example of static learning in the context of positioning. We assume the emission probabilities have been already initialised, where the way we initialise them will be explained later in Section 5.4. For simplicity here we only show some of the emission probabilities $b^m_{\text{C3}}(k)_t$, which control the likelihood of having location l_k in the measurements given that the actual state is location C3. Those initial emission probabilities are shown in the top row of Figure 5.5. We consider three probabilistic measurements as shown in Figure 5.4. Given the measurements observed, the new emission probabilities are evaluated by iteratively applying Equation (5.13), and the learned emission probabilities are shown in Figure 5.4. We can see that in this static case, the learned emission probabilities are clearly influenced by the measurements observed at time t: they actually reflect the uncertainty pattern in measurements (the typical way the measurements distribute their probability mass) reported by different systems. For instance, if the observed measurement is very uncertain (e.g. z_t^2 in Figure 5.4), the learned emission probabilities will become more uniformly distributed (see Figure 5.5(b)). Also given the state, we can observed that the learned emission probabilities tend to favour the locations that are assigned with more probability mass in the reported measurements (comparing the learned $b^1_{\text{C3}}(k)_t$, $b^2_{\text{C3}}(k)_t$ and $b^3_{\text{C3}}(k)_t$ in Figure 5.5 with the measurements z_t^1, z_t^2 and z_T^3 in Figure 5.4). This is expected, since the only information considered by static learning is the data observed at a single timestamp t.

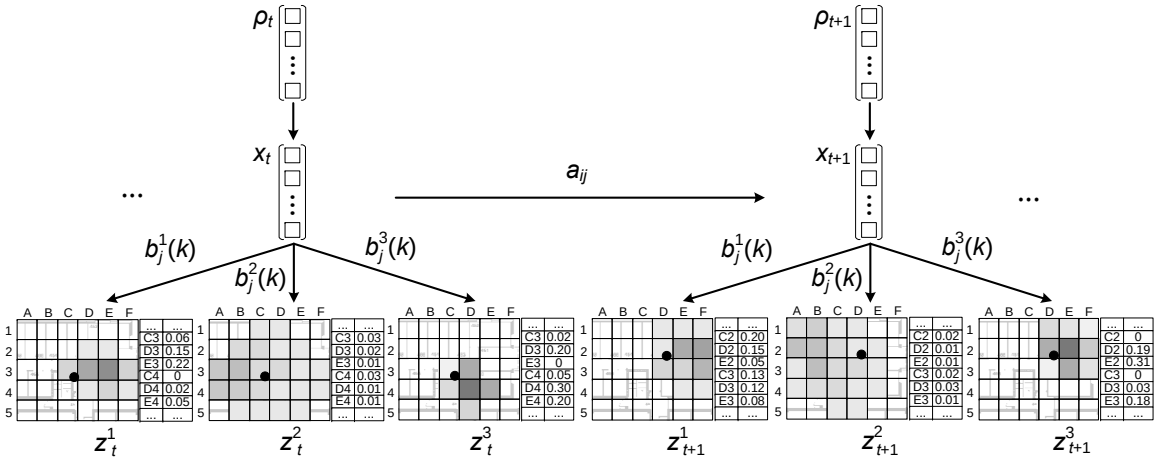


Figure 5.6: The model used in dynamic learning. At time t and $t+1$, the model emits multiple probabilistic measurements, each of which is a probability distribution over the locations. The current state is influenced by both the prior and the previous state, where the correlations are governed by the transition probabilities a_{ij}. The model parameters in this case are the transition probabilities a_{ij} and the emission probabilities $b_j^m(k)$, which are assumed to be time-invariant.

5.2.2 Dynamic parameter learning

We now consider dynamic parameter learning, which assumes the monitored states evolve over time and takes the full sequences of the measurements into account. We use the same setting as in dynamic inference, where the states x_1, ..., x_T form a stationary Markov process, i.e. $p(x_t|x_{1:t-1}) = p(x_t|x_{t-1})$, where the probability distribution $p(x_t|x_{t-1})$ governs how the current state transits to the next state. In the positioning context, the probability $P(x_t = l_j|x_{t-1} = l_i)$ is typically refereed to as the *transition probability* a_{ij}, which specifies the probability that the user moves from location l_i to l_j. We also assume that the starting point of the user, i.e. the initial state distribution $p(x_1)$, can be known exactly, e.g. from the card swipe at the main entrance when the user firstly enters the building.

Figure 5.6 shows the model used in dynamic learning. At timestamps t and $t+1$, the model emits multiple probabilistic measurements, each of which is a probability distribution over the locations. The state x_{t+1} is influenced by both the prior ρ_{t+1} and the previous state x_t, where the correlations are governed by the transition probabilities a_{ij}, as indicated by the directed edge between x_t and x_{t+1}. In this case, the model parameters that need to be learned are the transition probabilities a_{ij} and the emission probabilities $b_j^m(k)$ for each sensor system. Both a_{ij} and $b_j^m(k)$ are assume to be time-invariant (unlike static learning, here the parameters $b_j^m(k)$ are the same for all the timestamps). Existing learning algorithms, e.g. the Baum-Welch algorithm for standard hidden Markov models (HMMs), fail

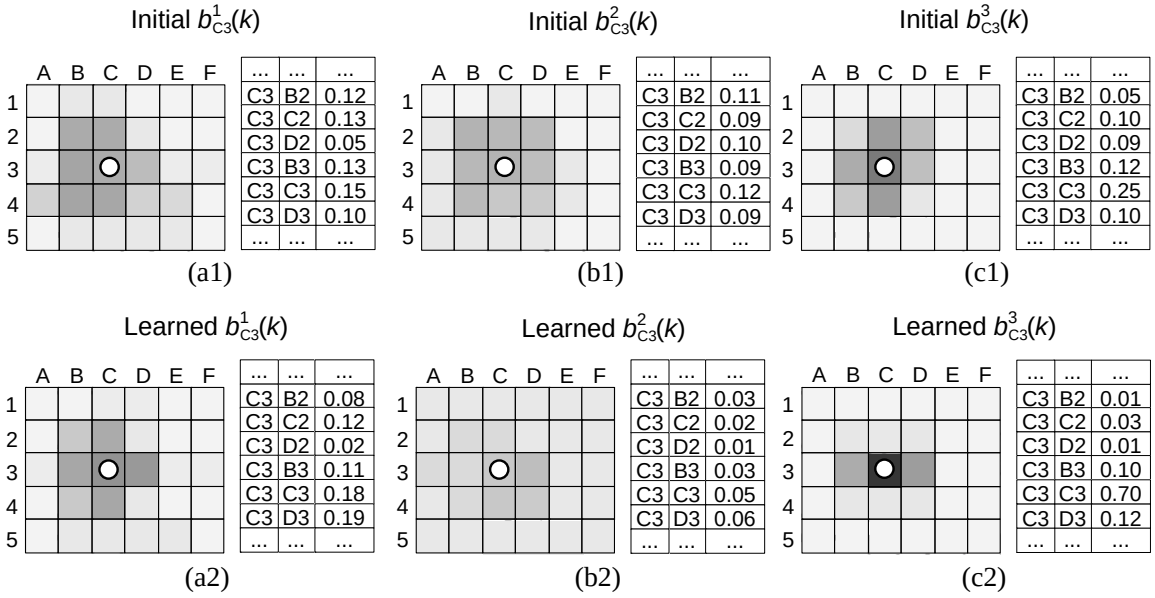


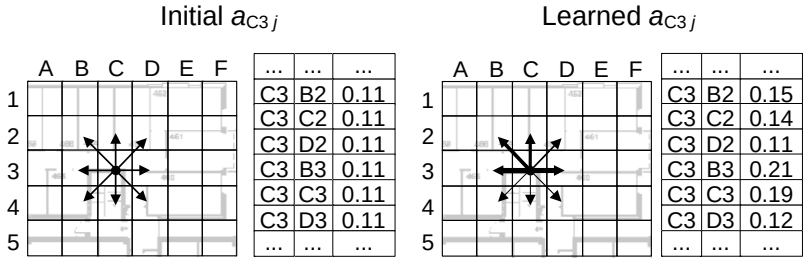
Figure 5.7: The initial emission probabilities of location C3 (on top) and the new emission probabilities learned by dynamic learning (on bottom). We can see that comparing to the parameters learned in static learning (shown in Figure 5.5), dynamic learning changes the emission probabilities significantly.

to work in this context because the emission probabilities $b_j^m(k)$ of our model have different meaning than those in standard HMMs (as explained in dynamic inference). Now we show how the proposed dynamic learning algorithm works.

Observed data: Unlike static learning, here the observed data O includes the whole sequences of probabilistic sensor measurements $z_{1:T}^{1:M}$, and priors $\rho_{1:T}$. At time t, a measurement z_t^m from the m-th sensor system is a vector $[z_t^m(1), ..., z_t^m(N)]$, where $z_t^m(k)$ is the belief of the system that the user is in location l_k at time t. Similarly, the distribution of the prior ρ_t is also a vector $[\rho_t(1), ..., \rho_t(N)]$, where $\rho_t(j)$ is the prior belief we possess that the user is at location l_j before observing any measurements. We denote the observed data as $O = \{z_{1:T}^{1:M}, \rho_{1:T}\}$.

Model parameters: In the dynamic case, the model parameters θ that need to be learned include both the state transition probabilities a_{ij} and the emission probabilities $b_j^m(k)$ for the coexisting sensor systems. We denote the parameters as $\theta = \{a_{ij}, b_j^m(k)\}$. The state transition probabilities a_{ij} represent the probabilities of transiting from a location l_i to another l_j. Unlike the standard HMMs, in our context the emission probabilities $b_j^m(k)$ for system sn_m represent the expected probability of having location l_k in the measurements of sn_m given the state is l_j.

Likelihood function: Given the observed data O and a state sequence $x_{1:T}$, the likelihood



(a) (b)

Figure 5.8: (a) The initial transition probabilities a_{C3j}, where transitions from location C3 to all the nine nearby locations have equal probabilities. (b) The new transition probabilities learned by dynamic learning. Now the transitions from location C3 reflect the movement patterns of the user: when at C3, she is more likely to move along the corridor.

function of the parameter θ is:

$$L(\theta; O, x_{1:T}) = P(\rho_{1:T}, x_{1:T}|\theta) \prod_{m=1}^{M} P(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta) \tag{5.15}$$

where the term $P(\rho_{1:T}, x_{1:T}|\theta)$ is the probability of the sequences of state $x_{1:T}$ and prior $\rho_{1:T}$, which can be determined by the current transition probabilities and priors ρ_t:

$$P(\rho_{1:T}, x_{1:T}|\theta) = \rho_1(x_1) \prod_{t=2}^{T} \rho_t(x_t) a_{x_{t-1}x_t} \tag{5.16}$$

where $a_{x_{t-1}x_t}$ are the transition probabilities between state x_{t-1} and x_t. $\rho_t(x_t)$ is the prior belief that the state is x_t. On the other hand, the term $P(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta)$ can be evaluated with the techniques discussed in dynamic inference (Section 4.1.2.3):

$$P(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta) = \prod_{t=1}^{T} \Big[\rho_t(x_t) \sum_{k=1}^{N} z_t^m(k) b_{x_t}^m(k) \Big] \tag{5.17}$$

where we sum over all possible locations l_k reported by the m-th sensor system with probability $z_t^m(k)$. $b_{x_t}^m(k)$ are the emission probabilities of having location l_k in the measurements of system sn_m given that the state is x_t.

***E*-step:** With the formulated likelihood function $L(\theta; O, x_{1:T})$, the *E*-step in each learning iteration evaluates the expected log function $Q(\theta', \theta)$ with respect to the new parameters θ' that need to be learned:

$$\begin{aligned} Q(\theta', \theta) &= E_{x_{1:T}|O,\theta}[\log L(\theta'; O, x_{1:T})] \\ &= \sum_{x_{1:T}} P(x_{1:T}|O, \theta) \Big[\log P(\rho_{1:T}, x_{1:T}|\theta') + \sum_{m=1}^{M} \log P(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta') \Big] \end{aligned} \tag{5.18}$$

where the terms $P(\rho_{1:T}, x_{1:T}|\theta')$ and $P(z^m_{1:T}|\rho_{1:T}x_{1:t}, \theta')$ can be further expanded as shown in Equations (5.16) and (5.17), with the new model parameters a'_{ij} and $b^m_j(k)'$. The term $P(x_{1:T}|O, \theta)$ is the posterior probability of a state sequence $x_{1:T}$ under the current model parameters, and can be evaluated with the dynamic inference technique discussed in Section 4.1.2.3.

***M*-step:** Given the current model parameters and the observed data, the derived $Q(\theta', \theta)$ is in fact a function of the new parameters a'_{ij} and $b^m_j(k)'$. As in static learning, we optimize the Q function with respect to each parameter, and the new parameters a'_{ij} and $b^m_j(k)'$ that maximise the Q function are given by:

$$a'_{ij} = \frac{\sum_{t=2}^{T} \alpha_{t-1}(i) a_{ij} \rho_t(j) r_t(j) \beta_t(j)}{\sum_{t=2}^{T} \hat{x}_{t-1}(i)} \tag{5.19a}$$

$$b^m_j(k)' = \frac{s^m_1(j,k) + \sum_{t=2}^{T} \Big[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} \Big] s^m_t(j,k) \beta_t(j)}{\sum_{t=1}^{T} \hat{x}_t(j)} \tag{5.19b}$$

where $\hat{x}_t(j)$ is the posterior probability that the state at time t is l_j. It is computed by the proposed dynamic inference algorithm as in Section 4.1.2.3. α_t and β_t are the *extended* forward and backward variables: $\alpha_t(j)$ represents the probability of observing all the probabilistic measurements from the M systems and priors until time t, and landing at state l_j, while $\beta_t(j)$ is the probability of have all the M sequences of future probabilistic measurements and priors from time $t+1$, given that the current state is l_j, as defined in Equation (4.22). $r_t(j)$ represents the probability of observing the M probabilistic measurements given the state is l_j, which is defined in Equation (4.21). Finally, $s^m_t(j,k)$ is the probability that the system sn_m has location l_k in measurement when state is l_j, as defined in the static case (Equation (5.14)). Note that all the above probabilities are evaluated with the current model parameters θ.

Now we explain the intuition behind the re-estimation formulas in Equation (5.19). The nominator in Equation (5.19a) is actually the total probability that the state transits from l_i to l_j during $t = 1:T$, while the denominator is the total probability that the state is l_i until time $T-1$. Therefore, the new transition probability a_{ij} is the expected number of transitions between location l_i and l_j. Similarly, the nominator in Equation (5.19b) is the total probability that l_k is reported by the system sn_m until time T when state is l_j, and the denominator is the total probability that the state is l_j. Thus the new emission probability is the expected probability with which location l_k is included in the measurements of sn_m

given the state is in l_j. Note that this is consistent with the ideas of the re-estimation formulas in standard HMMs: the new parameters are essentially re-calibrated by counting the number of transitions / emissions, divided by the number of times staying in a given state.

Complexity: As discussed above, at each iteration dynamic learning uses the proposed dynamic inference algorithm to evaluate the estimated state sequence $\hat{x}_{1:T}$, and thus the complexity of this algorithm is $O(ITMN^3)$, where I is the number of iterations, T is the number of timestamps, M is the number of sensor systems, and N is the number of discrete locations. Our algorithm is more expensive than the standard Baum-Welch algorithm which is $O(ITN^2)$, since we have M sequences of probabilistic observations rather than a single sequence of deterministic observations. However, as we motioned in Chapter 4, in practice we only have a few coexisting systems (M is $3 \sim 5$), and the probabilistic measurements (observed distributions) can be sparse. Therefore, with some optimisations our algorithm is not prohibitively expensive comparing to the Baum-Welch algorithm.

Example: We consider a similar example as in static learning. The top row of Figure 5.7 shows the initial emission probabilities of location C3 (i.e. the $b^m_{\text{C3}}(k)$) for three sensor systems sn_1, sn_2 and sn_3, which are identical to those in the static case. Figure 5.8(a) shows the initial state transition probabilities of location C3 (i.e. $a_{\text{C3}j}$), where transitions from location C3 to the nine nearby locations have equal probabilities. Figure 5.6 shows the probabilistic sensor measurements reported by the three systems at time t and $t+1$ respectively. Given the sequences of measurements, in each learning iteration the new transition probabilities are computed as in Equation (5.19a), while the new emission probabilities are evaluated as in Equation (5.19b). Figure 5.8(b) shows the learned state transition probabilities. We can see that dynamic learning reshapes the transition probabilities (note that the initial transition probabilities are uniform), so that they represent the typical movement patterns of the user (e.g. the user tends to move along the corridor rather than entering an office). Similarly, the bottom row of Figure 5.7 shows the learned emission probabilities of the three sensor systems sn_1, sn_2 and sn_3. We can observe that dynamic learning changes the emission probabilities significantly comparing to static learning (Figure 5.5). Also, the influence of measurements at one timestamp on the learned emission probabilities is smaller than static inference (see the measurements shown in Figure 5.6). This is because in the dynamic case we considered the full set of observed data, which provides more information on the behaviour of the model.

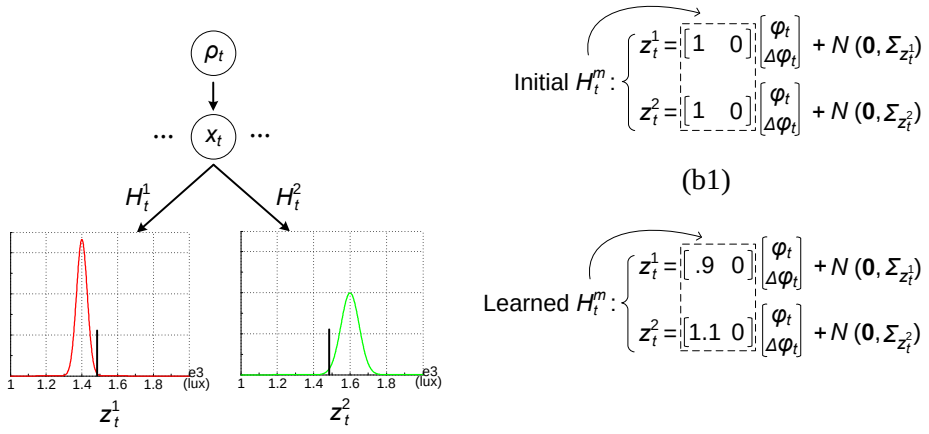


Figure 5.9: (a) The model with continuous variables used in static learning. At a given timestamp t, the state x_t is influenced by the prior ρ_t (here we assume uniform prior for simplicity), and emits multiple probabilistic sensor measurements (in this case two measurements z_t^1 and z_t^2). The model parameters are the measurement model H_t^m for sensor systems at each timestamp. (b) The initial and learned model parameters H_t^m. Note that here the state vector includes the real light intensity value ϕ_t and $\Delta\phi_t$, which represents the changing rate of light intensity (we will explain this in the dynamic case shortly). We can see that static learning can improve the initial parameters (which are trivial) based on the measurements observed in time t.

5.3 Parameter Learning for Continuous State Space

Now we consider the parameter learning problem for the case where the states and sensor measurements are continuous. We follow the same setting as in Section 4.2, and use the environmental monitoring scenario (introduced in Section 1.3.2) as the running example, where multiple sensor systems report light intensity measurements of an indoor environment. We assume that the sensor measurements are normally distributed: at a given timestamp t, the light intensity measurement z_t^m is Gaussian over $\mathbb{R}$: $z_t^m \sim \mathcal{N}(\mu_{z_t^m}, \Sigma_{z_t^m})$. We also assume that at certain timestamps we possess some prior knowledge on the state x_t, e.g. from another trusted sensing service or crowd sourcing application. We denote the prior knowledge on the state x_t as ρ_t, which is also Gaussian: $\rho_t \sim \mathcal{N}(\mu_{\rho_t}, \Sigma_{\rho_t})$.

As in the discrete case, here we also consider both the static and dynamic parameter learning, where the former learns the parameters with data observed at single timestamps, and the latter uses data observed over an entire sequence to re-estimate the model parameters. Now we explain how to extend the existing Kalman learner to re-calibrate the model parameters in both static and dynamic cases.

Static parameter learning: Static learning assumes no correlation between the monitored

states at different timestamps, and operates on slices of the observed data, i.e. the measurements and priors received at single timestamps t. Figure 5.9(a) shows the model used in static learning for the continuous case, where at time t, the state x_t is influenced by the prior ρ_t (as indicated by the directed edge from ρ_t to x_t), and emits multiple probabilistic measurement z_t^m, $1 \leq m \leq M$, each of which is Gaussian $\mathcal{N}(\mu_{z_t^m}, \Sigma_{z_t^m})$ over the sample space. As in static inference, the correlation between the measurements and the state is assumed to be linear: $z_t^m = H_t^m x_t + v_t^m$, where H_t^m is the measurement model.

However, unlike the standard linear dynamical systems which only accept deterministic measurements, in our model the variable v_t^m carries the uncertainty in the observed measurement z_t^m: $v_t^m \sim \mathcal{N}(0, \Sigma_{z_t^m})$. In this case, the observed data O_t at each timestamp is the M probabilistic sensor measurements $z_t^{1:M}$ from the coexisting sensor systems and prior on state ρ_t, i.e. $O_t = \{z_t^{1:M}, \rho_t\}$. The prior ρ_t is also assumed to be normally distributed, which represents the prior belief on the real light intensity x_t before any measurement is reported. The model parameters θ that need to be learned are the measurement models H_t^m at each timestamp t, which describe the measurements reported by sensor system sn_m given the actual state x_t. We initialise the parameters H_t^m with trivial values (as shown in Figure 5.9(b1), and the goal is to learn the new $H_t^{m\prime}$ that are most consistent with the observed data O_t.

As in static inference for the continuous case (discussed in Section 4.2), here we modify the standard Kalman learner by: a) using the variable v_t^m to carry the uncertainty in the observed measurements, and b) multiplying the prior distribution $p(\rho_t)$ to the estimated state before any measurement is incorporated. Then the new measurement model $H_t^{m\prime}$ in each learning iteration is given by (the detailed derivation is included in Appendix B.1):

$$H_t^{m\prime} = \frac{\mu_{z_t^m} \mu_{\hat{x}_t}^{\mathrm{T}}}{\mu_{\hat{x}_t} \mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t}} \tag{5.20}$$

where $\mu_{\hat{x}_t}^{\mathrm{T}}$ and $\Sigma_{\hat{x}_t}$ are the mean and covariance of the estimated state $\hat{x}_t$, computed by static inference under the current model parameter H_t^m. Note that the priors ρ_t are taken into account during the evaluation of the $\mu_{\hat{x}_t}$ and $\Sigma_{\hat{x}_t}$, and thus can bias the learned model parameters. Figures 5.9(b1) and (b2) show an example of the initial and learned model parameters H_t^m computed by this static learning algorithm. We can see that static learning can improve the initial parameters (which are trivial) based on the measurements observed in time t: it detects that measurements from system sn_1 typically underestimate the real state x_t, while measurements from sn_2 overestimate x_t.

Dynamic parameter learning: Dynamic learning assumes the states are temporally correlated, and learns the model parameters with all the observed measurements and priors.

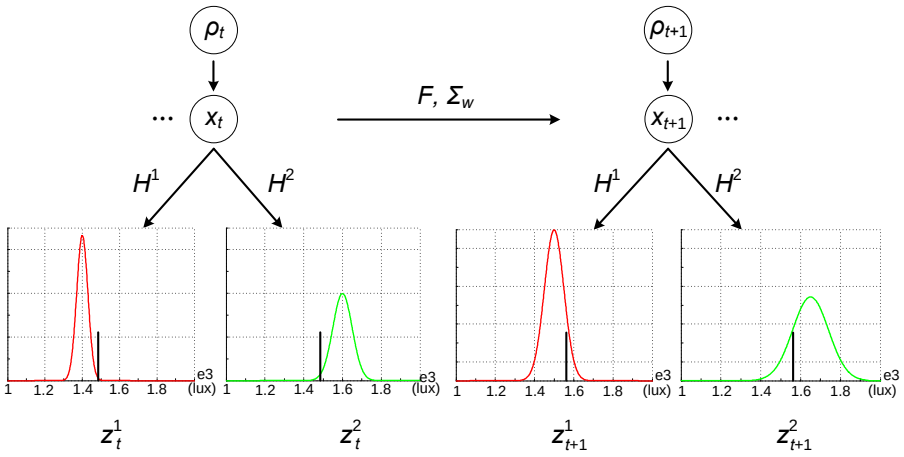


Figure 5.10: The model with continuous variables used in dynamic learning. At a given timestamp $t+1$, the state x_{t+1} is influenced by both the prior ρ_{t+1} and the previous state x_t, and emits multiple probabilistic sensor measurements (in this case two measurements z_{t+1}^1 and z_{t+1}^2). The model parameters are the measurement model H^m for each sensor system, the transition model F and the covariance of the process noise Σ_w.

Figure 5.10 shows the model used in dynamic learning. At a given timestamp $t+1$, the state x_{t+1} is influenced by both the prior ρ_{t+1} and the previous state x_t, and emits multiple probabilistic sensor measurements. As in dynamic inference, we assume the correlation between the states at different timestamps are linear. In the example of light intensity monitoring, we assume the state vector contains two elements: the actual light intensity value ϕ_t, and the current changing rate, i.e. velocity of the light intensity $\Delta\phi_t$. Then the state vector at time $t+1$ is given by: $\begin{pmatrix} \phi_{t+1} \\ \Delta\phi_{t+1} \end{pmatrix} = F_{t+1} \begin{pmatrix} \phi_t \\ \Delta\phi_t \end{pmatrix} + w_{t+1}$, where F_{t+1} is the transition model, and w_{t+1} is a vector of the process noise. As in the static case, we also assume that a measurement z_t^m observed at timestamp t linearly depends on the state x_t: $z_t^m = H_t^m x_t + v_t^m$, where H_t is the measurement model, and v_t^m carries the uncertainty in the observed measurement z_t^m.

In dynamic case, the observed data O includes the M sequences of the probabilistic measurements $z_{1:T}^{1:M}$, and the sequence of priors $\rho_{1:T}$, i.e. $O = \{z_{1:T}^{1:M}, \rho_{1:T}\}$. The model parameters θ that need to be learned include three elements: the transition model F_t, the covariance of the process noise vector Σ_{w_t}, and the measurement model H_t^m. F_t maps the current state vector to the next, Σ_{w_t} governs the inflated uncertainty in the state vector during state transitions, and H_t^m controls how the measurements of the m-th sensor system are correlated with the actual states. Since dynamic learning operates on the full sequences of observed data, here for simplicity we assume the model parameters F_t, Σ_{w_t} and H_t^m are independent of time t, and denote them as F, Σ_w, and H^m.

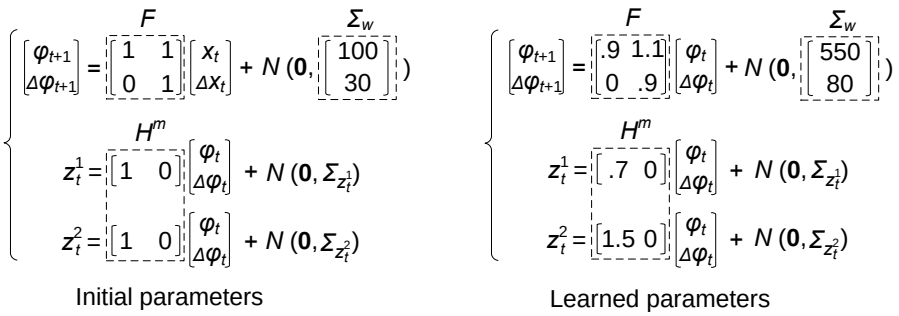


Figure 5.11: (a) The initial and learned model parameters F, Σ_w and H^m. The state vector includes ϕ_t and $\Delta\phi_t$, which represent the actual light intensity and its changing rate. We assume the next state x_{t+1} is a function of the current state x_t, while the measurements z_t^m only depend on the current state x_t. We can see that dynamic learning changes the initial parameters significantly, and detects that a) system sn_1 consistently underestimates the real state; b) system sn_2 typically overestimates the real state; and c) the initial process noise for both ϕ_t and the velocity $\Delta\phi_t$ is underestimated.

As in the static case, we modify the standard Kalman learner, where we use v_t^m to represent the uncertainty in the observed measurement, and consider the priors ρ_t at each timestamp. Then the new parameters F', Σ_w', and $H^{m\prime}$ in each learning iteration as follows (the detailed derivation is included in Appendix B.1.1):

$$F' = \frac{\sum_{t=1}^{T-1}(\mu_{\hat{x}_{t+1}}\mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t})}{\sum_{t=1}^{T-1}(\mu_{\hat{x}_t}\mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t})} \tag{5.21a}$$

$$\Sigma_w' = \frac{1}{T-1}\sum_{t=2}^{T}[\mu_{\hat{x}_t}\mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t} - F'(\mu_{\hat{x}_t}\mu_{\hat{x}_{t-1}}^{\mathrm{T}} + \Sigma_{\hat{x}_t})^{\mathrm{T}}] \tag{5.21b}$$

$$H^{m\prime} = \frac{\sum_{t=1}^{T}\mu_{z_t^m}\mu_{\hat{x}_t}^{\mathrm{T}}}{\sum_{t=1}^{T}(\mu_{\hat{x}_t}\mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t})} \tag{5.21c}$$

Here $\mu_{\hat{x}_t}$ and $\Sigma_{\hat{x}_t}$ are the mean and covariance of the estimated state $\hat{x}_t$, which is computed by using dynamic inference as discussed in Section 4.2, under the current model parameters F, Σ_w, and H^m. The priors ρ_t have been taken into consideration during the evaluation of $\mu_{\hat{x}_t}$ and $\Sigma_{\hat{x}_t}$, and thus can bias the learned model parameters.

The above parameter re-estimation formulas actually share the same intuition with that in the discrete case. The new transition model F' can be viewed as the expected number of transitions from current state x_t to the next state x_{t+1}, divided by the total number of times

of being in the current state x_t, while the new measurement model $H^{m\prime}$ for the m-th sensor system is in fact the number of times of being at state x_t and observing measurement z_t^m, divided by the total number of times of being in state x_t. Figures 5.11(a) and (b) show the initial and learned parameters, computed by this dynamic learning algorithm. We can see that dynamic learning changes the initial parameters significantly, and is able to detect that: a) system sn_1 consistently underestimates the real state; b) system sn_2 typically overestimates the real state; and c) the initial process noise for both ϕ_t and the velocity $\Delta\phi_t$ is underestimated.

5.4 Implementation Details

In this section, we discuss several important implementation details of the proposed learning-based accuracy estimation approach.

Initialisation of the model parameters: As discussed above, the general Expectation Maximisation (EM) scheme only converges to a local optimum, and there is no guarantee that the globally optimum can be reached. One important issue is how to initialise the model parameters so that the EM scheme is likely to produce the global optimal estimates of the parameters. There are various existing techniques of selecting initial model parameters in state space models [2]. This thesis considers a simple approach that produces the initial model parameters using the observed measurements and certain physical constraints. We explain the techniques in the context of the discrete case (positioning scenario), noting that they also apply to the continuous case. In our experiments, the indoor environment is divided into blocks, where each block represents one discrete location. For the initial state transition probabilities $a_{ij_{\text{init}}}$, we define them as follows: if the user is currently in location l_i, then she has equal probabilities (we actually add tiny random noise to those probabilities to avoid tie situations) of transiting to the nine adjacent blocks (including l_j itself). Note that here we implicitly assume that the user is not able to move very fast.

To evaluate the initial emission probabilities $b_j^m(k)_{\text{init}}$, we consider all measurements reported by system sn_m. For a measurement z_t^m, we find the location $\check{l}$ that has the largest reported probability and consider it as the pseudo ground truth. Then $b_j^m(k)_{\text{init}}$ is computed as the average of the probabilities $z_t^m(k)$ for all measurements from sn_m, whose $\check{l}$ is l_j. In other words, for a sensor system sn_m we assume that the real states are $\check{l}$ and use them to initialise the parameters $b_j^m(k)$ empirically.

It is also worth pointing out that the inference-based approach discussed in Chapter 4 also uses the above procedure to initialise the model parameters. However, different from the learning-based approach proposed in this chapter, the inference-based approach directly uses those parameters to infer the latent states, without further re-estimating them with the observed data.

Insufficient training data: Since the observed data is always finite, another issue of the learning-based approach is how to deal with the problem of insufficient training data, especially for the static learning approach. In our experiments we only have few coexisting systems (three in the discrete case and two in the continuous case), and therefore the learned parameters from static learning can be biased. In our implementation of static learning, we address this with the following method. Consider the indoor positioning example, where the model parameters in static learning are the emission probabilities $b_j^m(k)$. As discussed above, the initial values $b_j^m(k)_{\text{init}}$ are evaluated by counting the expected number of times that location l_k is reported and the pseudo ground truth $\check{l}$ is location l_j, divided by the total number of times when the pseudo ground truth $\check{l}$ is location l_j. Therefore at the beginning of the learning process, we set the parameters $b_j^m(k)^{(0)}$ as $b_j^m(k)_{\text{init}}$. Then in the i-th learning iteration, we firstly compute the emission probabilities $b_j^m(k)'$ based on the observed data as in Equation (5.13), and the new emission probabilities $b_j^m(k)^{(i+1)}$ for the next learning iteration are evaluated as a weighted average of the current emission probabilities $b_j^m(k)^{(i)}$ and the computed $b_j^m(k)'$: $b_j^m(k)^{(i+1)} = b_j^m(k)^{(i+1)} + c\, b_j^m(k)'$, where c is the innovation factor (in our experiments, we use constant values based on the size of the observed data set). In this way, the contribution from the learned model parameters (evaluated with data observed in single timestamps) is controlled by the factor c. This can prevent abrupt changes of the model parameters caused by insufficient training data, and provides a smooth improvement in parameter estimation.

Over-fitting: Finally, a common problem for all learning techniques is over-fitting. The Expectation Maximisation (EM) scheme used in this thesis is an iterative approach, where in each learning iteration it finds the parameters that increase the likelihood of the observed data. However, this can cause problems because a) the parameters that have higher likelihood may not necessarily be closer to the real ones, and b) given the limited amount of observed data, the parameters learned on this dataset may not work on others. We observe this over-fitting problem in our experiments, and here we explain the problem in the indoor positioning scenario. Figure 5.12(a) shows the accuracy estimation performance (under the proximity-based accuracy metric f_ϵ^p) of dynamic learning as we increase the number of learning iterations. The x axis is the number of iterations, while the y axis is the *accuracy estimation error*, which is defined as the squared difference between the estimated accuracy

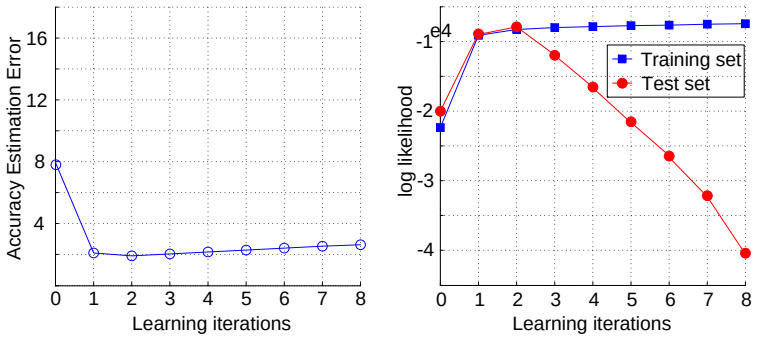


Figure 5.12: (a) The performance of accuracy estimation in dynamic learning, as we increase the number of learning iterations (in the positioning scenario). (b) Log likelihood of the observed data (both training and test) as the number of learning iterations increases.

and the real accuracy evaluated with the knowledge of the ground truth (we will further explain this metric in the next chapter). As we can see, the error in accuracy estimation first drops sharply at the beginning, but then starts to increase as more learning iterations are performed. Note that smaller estimation error means better accuracy estimation performance: the estimated accuracy is closer to the real accuracy. This clearly indicates the over-fitting problem, and in this thesis, we address this by randomly selecting a subset of the observed data as a training set that we use for learning, and another subset of data as a test set that we used to detect when to terminate the learning process. Then we run learning process on both the training and test sets simultaneously, and terminate the learning process when the learned parameters decrease the likelihood of the test set. As shown in Figure 5.12(b), we terminate learning in the third iteration when the log likelihood of the test set starts to decrease, which can effectively prevent the problem of over-fitting.

5.5 Discussion

In this chapter, we have proposed a novel learning-based accuracy estimation approach, which employs parameter learning techniques to improve estimating the accuracy of coexisting sensor systems. We have proposed two different variants of the learning-based approach, depending on the accuracy requirements of the application. The first variant works in general cases, where the sensing application seeks to index the accuracy of the sensor systems over non-state attributes. In this case, the proposed learning-based approach implements the four-layer accuracy estimation framework proposed in Chapter 3. However,

unlike the inference-based approach discussed in the previous chapter, in the state estimation layer the proposed learning-based approach firstly re-estimates the model parameters based on the observed measurements and priors, and then uses the learned parameters to infer the latent states. On the other hand, the second variant is designed for the cases where the accuracy of sensor systems needs to be indexed over the monitored states, which cannot be addressed by the inference-based approach. In this case, the proposed learning-based approach skips the state estimation layer, merges the accuracy estimation and indexing layers, and derives the indexed accuracy from the learned model parameters directly.

We have shown that for both variants of the proposed learning-based approach, the key step is to learn the model parameters that are the most consistent with the observed data. Existing parameter learning algorithms, such as the Baum-Welch algorithm [135] for standard hidden Markov models fail to work in our context, since they only consume one deterministic observation at a timestamp, and do not consider prior knowledge on states other than at the beginning. We have proposed new parameter learning algorithms based on the Expectation Maximisation (EM) scheme for both discrete and continuous cases, which can accept probabilistic measurements from multiple sensor systems and incorporate prior knowledge on arbitrary states to improve parameter estimation. We have also proposed two different types of parameter learning algorithms, static learning and dynamic learning, depending on whether we exploit the temporal correlations between the states in the learning process. We have shown that static learning can re-estimate the model parameter based on data observed in single timestamps, while dynamic learning considers the states as a stochastic process, and learns the parameters with the full set of measurements and priors.

We have shown that static learning can improve the estimation of model parameters, but the improvement is limited, due to the fact that it only considers the measurements and prior observed at a single timestamp. On the other hand, dynamic learning changes the model parameters significantly, since the new parameters are learned with the full sequence of observed data, which can provide more information on the behaviour of the model. We have also discussed several important implementation issues of the proposed learning approach, and provided techniques to address the problems of insufficient training data and over-fitting.

Compared to the inference-based approach, the learning-based approach has its own merits. First of all, it does not rely on any prior knowledge of the model parameters: even the initial parameters are trivial (as we discussed in Section 5.3), the learning-based approach can recover useful knowledge from the observed data to learn better parameters. Secondly, when we possess some prior knowledge on the model parameters (e.g. in the

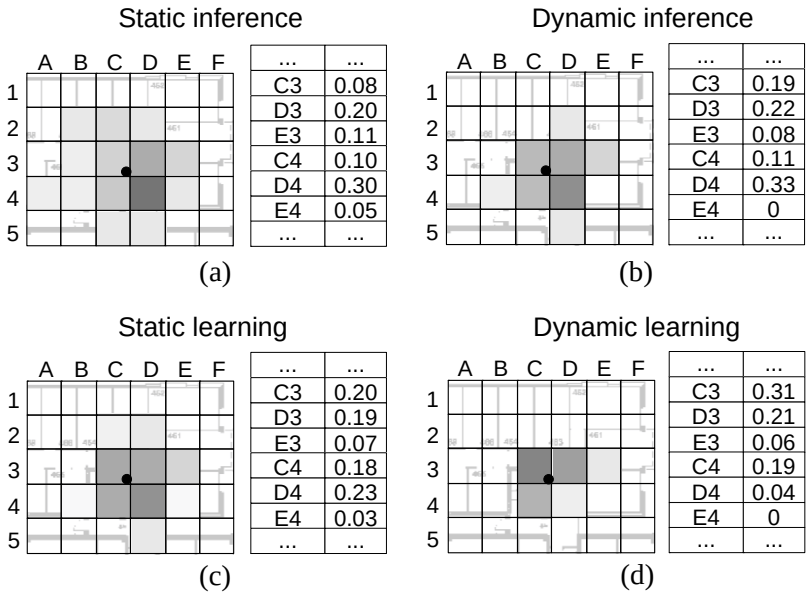


Figure 5.13: The estimated state at time t computed by different algorithms in the positioning scenario, where the models and measurements at time t are shown in Figures 5.4 and 5.6.

positioning scenario the initial emission probabilities are estimated from the reported measurements), the learning-based approach can iteratively refine the model parameters, and thus improve the estimation of states. Figure 5.13 shows an example of the estimated states computed by different algorithms in the positioning scenario. The static and dynamic models and measurements at time t are shown in Figures 5.4 and 5.6. We can see that static inference fuses measurements from multiple sensor systems and can produce reasonable state estimate at time t. Dynamic inference improve the state estimate, and assigns more probability to the location C3 which is the ground truth. Static learning further refines the estimate state, and reduces the probability assigned to location D4 which is actually far away from the ground truth. Dynamic learning produces the best state estimate, where the ground truth location C3 and the locations nearby (D3 and C4) are assigned with the majority of the probability mass. Finally, the inference-based approach fails to work when the sensing application seeks to index the accuracy over monitored states, while the learning-based approach can address this by deriving accuracy from the learned model parameters directly.

However, the learning-based approach is inherently more expensive than inference. This is because learning typically involves multiple runs of inference when evaluating the expected likelihood function (the Q function). In the next chapter, we will perform a systematic experimental evaluation of both the inference- and learning-based approaches on

two real sensor datasets, and compare them extensively in terms of accuracy estimation and computational cost.

Chapter 6

Evaluation

This chapter provides a systematic experimental evaluation of the proposed accuracy estimation approaches in the context of two real-world sensing applications. We introduce our experiment setup in Section 6.1, which includes two different sensing scenarios: an indoor positioning scenario and an indoor environmental monitoring scenario. We show how we build the research tesbeds, collect the sensor readings and pre-process them for further experiments. Then in Section 6.2 we explain the competing algorithms and the metrics against which the algorithms are evaluated. Section 6.3 presents the results of our experiments, and shows that the proposed approaches outperform the competing ones in both scenarios, and different approaches have their own merits in different contexts. We conclude this chapter with further discussions on the experimental evaluation in Section 6.4.

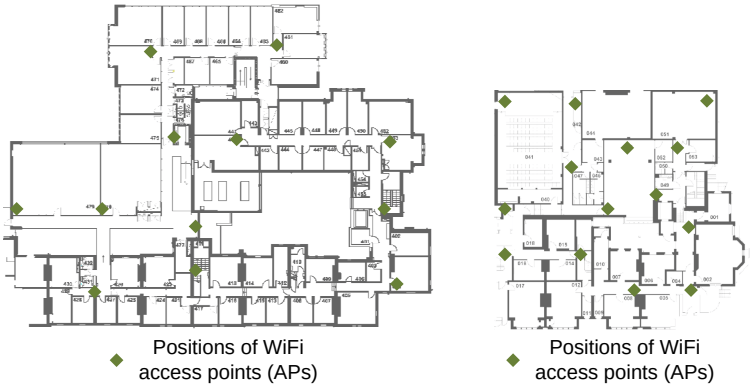
6.1 Experiment Setup

We consider two different experiment scenarios, one is an indoor positioning scenario and the other is an indoor environmental monitoring scenarios. In both scenarios, data is collected from hardware deployments, where multiple sensor systems run in parallel and monitor certain physical signals. In the following text, we explain the experiment setup of the two scenarios in more detail, including the sensor deployments, data collection procedures and pre-processing techniques.

6.1.1 Indoor positioning scenario

6.1.1.1 Testbeds

We have built two indoor positioning testbeds in a multi-storey office building:



(a) (b)

Figure 6.1: (a) The 4th floor (4F) testbed, which has 12 WiFi access points deployed. (b) The basement testbed (0F), which has 14 WiFi access points deployed.

The 4th floor (4F) testbed: The first testbed is built on the 4th floor of the Department of Computer Science, University of Oxford. This testbed is a typical office environment, with a few meeting rooms and kitchens. The dimensions of the 4th floor are approximately $65m \times 45m$, and it is mainly occupied by research staff and students. We have deployed a total number of 12 WiFi access points (APs) at different locations of the floor, which periodically broadcast WiFi beacons. Figure 6.1(a) shows the floor plan of the testbed and the deployment locations of the WiFi APs (we will explain shortly how the APs are used by different sensor systems).

The basement (0F) testbed: The second testbed is built in the basement of the Department of Computer Science, University of Oxford. This testbed is smaller than the 4th floor, and includes mainly seminar rooms, lecture theaters and other common areas. The size of the testbed is approximately $30m \times 40m$. We have deployed a total number of 14 WiFi APs at different positions of the floor, which also broadcast WiFi beacons at fixed time intervals. Figure 6.1(a) shows the floor plan of the testbed and the deployment locations of the WiFi APs.

6.1.1.2 Sensor systems

We have built and tested multiple indoor positioning systems based on two different positioning modalities: a) the WiFi beacons sent by the deployed access points, which are then picked up by the mobile devices carried by the users, such as cellphones or tablets; and b) the inertial information received from the inertial measurement units (IMU), which are mounted to the feet of the users. Figure 6.2 shows a WiFi access point and a foot-mounted IMU used in our experiments. As shown in Figure 6.2(a), in our experiments we use an-

(a) (b)

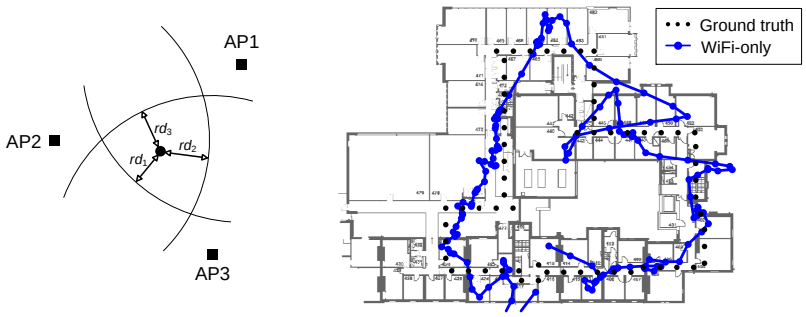
Figure 6.2: (a) A WiFi access point used in our experiments, which is an android phone (Huawei U8160) attached to the wall. (b) An IMU used in our experiments, which is mounted to one foot of a user.

droid phones (Huawei U8160) attached to the walls as WiFi access points, whose WiFi hardware is configured to ad-hoc mode by an application (with root access). The WiFi channels of the phones are set to 11 to avoid potential signal interference with the existing WiFi infrastructure in the building (APs for OWL and eduroam networks), which mainly operate on channels 1 and 6.

The foot-mounted IMU used in our experiments are the off-the-shelf product x-IMU from a UK-based company x-io [140]. The x-IMU contains a rich set of on-board sensors, including a 16-bit gyroscope, a 12-bit accelerometer, a 12-bit magnetometer and a thermometer. It runs its own IMU and AHRS algorithms [141] to provide real-time inertial measurements (i.e. the position and orientation changes of the IMU in 3D space) at variable data rates up to 512 Hz. The x-IMU also comes with an on-board SD card for data storage, and can stream the measured inertial data via USB or Bluetooth connections. In our experiments, we attach the x-IMUs to the feet of the users as shown in Figure 6.2(b), and set the data rate to 128 Hz. The generated data are stored in the SD card and retrieved later for processing, to extend the battery life.

Based on the positioning modalities discussed above, we have built the following three classes of indoor positioning systems: the WiFi-only systems that only use WiFi beacons, the IMU-only systems that only use the inertial measurements, and the WiFi+IMU systems that combine the two modalities.

WiFi-only systems: The WiFi-only systems estimate the positions of a user solely based on the WiFi beacons broadcast by the deployed WiFi access points (APs). A WiFi-only system owns a subset of the APs, and receives WiFi beacons from those APs when the mobile devices carried by the users are close to them. In our experiment, a received beacon is a triple $\langle t, \text{SSID}, \text{RSSI} \rangle$, where t is the timestamp when this beacon is received. We



(a) (b)

Figure 6.3: (a) WiFi signal strength triangulation with three access points AP1, AP2, and AP3 (black squares). The estimated location is indicated by the black dot, which minimises the residual squared distance $(rd_1)^2 + (rd_2)^2 + (rd_3)^2$. (b) A trajectory estimated by the WiFi-only system with all the 12 APs (as shown in Figure 6.1(a)), where the black dots indicate the ground truth.

assume that all the access points and mobile devices are time synchronised beforehand, and in our experiments the WiFi beacons are received at a rate of approximately 1$\sim$2 Hz. The SSID indicates the origin of the beacon (i.e. the AP that has emitted the beacon), and RSSI is an integer ranging from -90 to 0, representing the received signal strength of this beacon. Typically at a given timestamp t, a mobile device of a user can receive multiple beacons from different access points, and the WiFi-only systems determine the positions of the user using signal strength triangulation. This is a widely used technique, and in our experiments we implement the WiFi signal strength triangulation based on our work in [95]. The WiFi triangulation has two steps: 1) given a received beacon, calculate the distance between the receiver (i.e. the mobile device carried by the user) and the access point who has sent this beacon; and 2) with the calculated distances with three or more access points, estimate the position of the user.

For the first step, the relationship between the received signal strength of a beacon and the distance with the AP can be described by the propagation model in the indoor environment, which depends on many factors of the indoor environment, such as the walls, the structure of the floor, the number of windows, etc. In our experiments, we consider a simple log-normal indoor propagation model:

$$P(d)[dBm] = P(d_0) + 10\gamma \log \frac{d}{d_0} + WAF + X_\sigma \tag{6.1}$$

where $P(d)$ is the received signal strength value (in dBm) in a location that is d meters away from the access point, and d_0 is the reference distance (say $1m$). γ is the distance power loss coefficient, WAF is the wall attenuation factor, and X_σ is a white noise with variance σ. This model explains how the received signal strength value varies over distance.

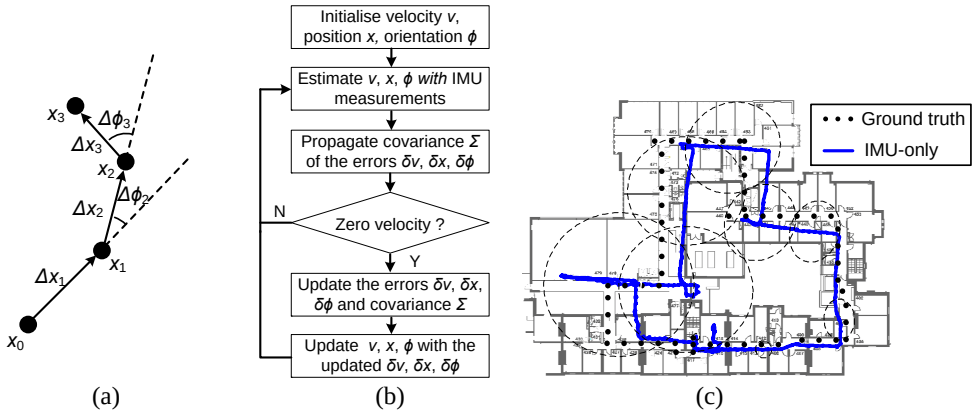


Figure 6.4: (a) The basic idea of inertial navigation is to estimate the current position based on the cumulated changes in position and orientation. (b) Workflow of the pedestrian dead reckoning (PDR) algorithm implemented in our experiments. (c) The trajectory generated by an IMU-only system, where the dashed ellipses indicate the uncertainty in the position measurements.

Therefore, given a received beacon, one can estimate the distance between the user and the AP that emits this beacon based on the received signal strength. In our experiments, the parameters of the propagation model, i.e. $P(d_0)$, γ, WAF and σ are learned through a training phase before doing the experiments over the entire testbed.

For the second step, the WiFi-only systems estimate the user position with the estimated distances to at least three access points. Figure 6.3(a) shows an example of WiFi triangulation used in our implementation (with three APs). As shown in Figure 6.3(a), given the estimated distances to the access points AP1, AP2 and AP3, one can draw three circles with those distances (Figure 6.3(a) only shows parts of the circles). In our implementation, the WiFi-only systems find the optimal position that minimises the overall residual squared distance $(rd_1)^2 + (rd_2)^2 + (rd_3)^2$ to all the borders (indicated by the black dot in Figure 6.3(a)). Figure 6.3(b) shows an estimated trajectory of a WiFi-only system deployed on the 4th floor testbed. We can see that the trajectory produced by this system is very noisy, and without any uncertainty information associated with the estimated positions.

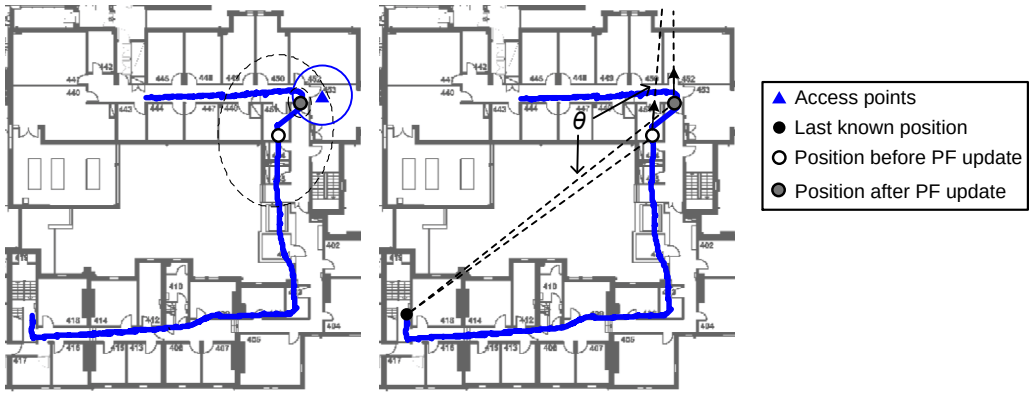
IMU-only systems: The IMU-only systems estimate the position of the user only based on the inertial measurements from the IMUs mounted on the feet of the user. In our experiments, the IMU-only systems implement the pedestrian dead reckoning (PDR) algorithm in [103] to estimate positions. With a known starting point, conceptually the PDR algorithm estimates the current position of the user by tracking the relative displacement and angle changes of the user's feet. The PDR algorithm has two important components: a) the inertial navigation (IN); and b) the zero-velocity updating (ZV). In the following text we briefly explain how the PDR algorithm works in our context.

The inertial navigation (IN) evaluates the feet position of the user by considering the acceleration and angular velocity measurements reported by the accelerometer and gyroscope respectively. For acceleration measurements, IN subtracts the gravity from the vertical axis, and integrates the measured acceleration twice to get the velocity and the displacement from the previously known position. Similarly, IN integrates the measured angular velocity reported to calculate the orientation changes. Figure 6.4(a) shows the basic idea of inertial navigation (IN). At each timestamp, IN calculates the position changes Δx_t and the orientation changes $\Delta \phi_t$ from the IMU measurements. Given the known initial position x_0, the current position and orientation can be evaluated based on the accumulated position and orientation changes.

However, due to the noise in the measurements, the position estimates produced by IN can drift several meters in just a few seconds. The second component of the PDR algorithm, the zero-velocity updating (ZV) addresses this by exploiting human gait to clamp the odometric drift to a reasonable level. The idea of the zero-velocity update is that when one foot hits the ground during walking, which is often refereed to as the *stance phase*, its velocity should be zero. If the estimated velocity is not zero, the drift in velocity is known. Since the errors in velocity are correlated with the errors in both the orientation and position, this zero-velocity observation can be used to update the current estimation of orientation and position as well.

In our experiments, the PDR algorithm is implemented with a complementary extended Kalman filter (EKF), where the states include the errors in velocity, orientation and position. Figure 6.4(b) shows how our implementation of the PDR algorithm works. We initialise the EKF with the velocity $v_0 = 0$, a known starting position x_0 and the initial orientation ϕ_0 which is computed from the magnetometer readings. In each iteration, the algorithm estimates the current velocity v, position x and orientation ϕ with the inertial measurements, as explained above. It also propagates the covariance Σ of the states (the errors in velocity, position and orientation δv, δx, $\delta \phi$) as in normal EKF. Then the algorithm decides whether there is a zero-velocity update by checking the gyroscope reading: if the angular velocity of the foot is below a small threshold, the foot is considered to be in stance phase, and the algorithm needs to perform a zero-velocity update. During the zero-velocity update, the algorithm updates the states, i.e. the errors δv, δx, $\delta \phi$, and the covariance Σ in the errors, and then uses the updated errors to correct the current estimate in velocity, position and orientation for the next filtering iteration. On the other hand, if there is no zero-velocity update, the algorithm just proceeds to the next iteration.

Figure 6.4(c) shows a trajectory generated by an IMU-only system in our experiments. The ellipses show the estimated covariances of the position estimates. As the user moves



(a) (b)

Figure 6.5: (a) The position update step. The particle filter samples according to the current position estimate, and weights the particles with respect to the signal strength value of the received WiFi beacon. (b) The orientation update step. The orientation of the user is updated based on the last known position.

counterclockwise, we can see that the position estimates are very accurate at the beginning (respect to the ground truth), but start to drift quite heavily after several turns. Note that the estimated covariances towards the end of this trajectory (the left part) are very large, which means that the system is very uncertain about the estimated positions.

WiFi+IMU systems: As discussed above, both of the WiFi-only and IMU-only systems have their drawbacks: the WiFi-only system typically can not produce accurate trajectories (as shown in Figure 6.3(b)), while trajectories estimated by the IMU-only systems drift heavily after a short period of time (as shown in Figure 6.4(c)). Therefore in our experiments, we designed a new class of positioning systems, which fuse the inertial measurements from the foot-mounted IMUs and the WiFi beacons from the access points to estimate the position of the user. The idea of our implementation is similar with existing work in [142], which incorporates WiFi signal strength measurements in pedestrian dead reckoning (PDR) based SLAM.

The main idea of our implementation is that we use the received signal strength of the WiFi beacons to correct both the position and orientation estimates produced by the PDR algorithm. The signal strength of a beacon can be converted to the approximate distance between the user and the AP who has sent this beacon, with the propagation model as explained in the WiFi-only systems. This indicates the possible positions of the user, and can be used to update the estimates in the filtering process of the PDR algorithm. In our implementation, we do not directly feed this to the EKF in the PDR algorithm because the state in the EKF is the *errors* in velocity, position and orientation, while a signal strength measurement is essentially the distance to a known point (the AP position). Instead, we

use another particle filter on top of the PDR algorithm, which treats the output of the PDR algorithm and the received WiFi beacons as measurements, and updates the position and orientation of the user accordingly.

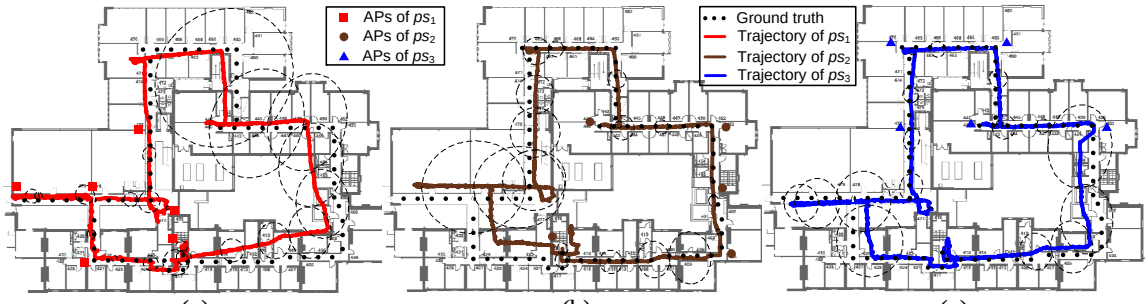
When the user receives a WiFi beacon with strong enough signal strength (in practice we find that the propagation model we use is more accurate for larger signal strength values), the WiFi-IMU system works in two steps: 1) it updates the estimated position with the particle filter; and 2) it updates the estimated orientation with the last known position and the current estimated position. In the first step, the particle filter first samples the current estimation of the position. Then it weights the generated particles according to their distances to the known AP position. Those particles whose distances to the AP are more consistent with the distance estimated from the actual received signal strength have larger weights. Then the filter re-samples the weighted particles to estimate the current position and the uncertainty associated with it. Figure 6.5(a) shows an example of this step.

In the second step, we use a simple approach for orientation update. As shown in Figure 6.5(b), we connect the estimated positions before and after particle filter update with the last known position (e.g. the position updated by the previous WiFi beacon). We calculate the angle θ between the two lines, and offset the current orientation by θ, which essentially "rotates" the drifted orientation back on track.

The WiFi+IMU systems work well in our settings, and can produce reasonable indoor positioning services, which are consistently superior than the WiFi-only and IMU-only systems. Therefore in our experiments, we choose the WiFi+IMU systems as the research vehicles to evaluate the proposed accuracy estimation approaches. We create different WiFi+IMU systems, each of which has access to a subset of the WiFi access points in different locations of the floor. The systems estimate positions of the user by combining the inertial measurement and the WiFi beacons received from the APs they own, as discussed above. Figure 6.6 shows three different WiFI-IMU systems ps_1, ps_2 and ps_3 used in our experiments, and the example trajectories generated by them.

6.1.1.3 Data collection and pre-processing

Data collection: We run multiple WiFi+IMU systems in parallel on both the 4th floor and basement testbeds, and track two different users (research students). Two x-IMUs are attached to the feet of the users, and they carry two different mobile devices, a Nexus S and an Asus TF201 tablet with them all the time. For the 4th floor testbed, we create 4 different WiFi+IMU positioning systems ps_1, ps_2, ps_3 and ps_4 (example trajectories generated by $ps_1 \sim ps_3$ are shown in Figure 6.6), and track the users for 20 days. For the basement testbed, we create 3 different WiFi+IMU systems ps_5, ps_6 and ps_7, and track the users for



(a) (b) (c)

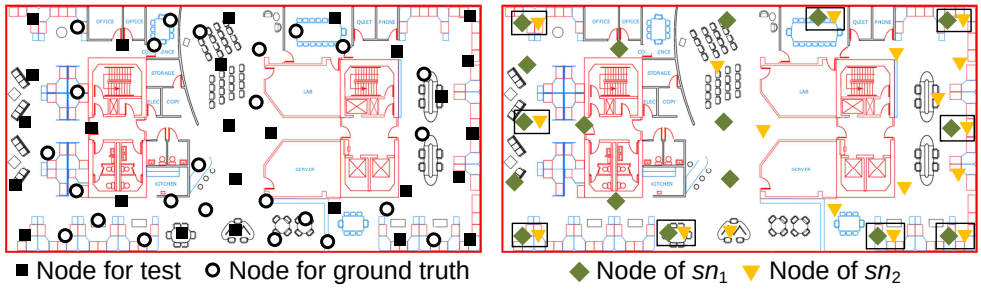
Figure 6.6: Three different WiFi+IMU positioning systems ps_1, ps_2 and ps_3 deployed on the 4th floor testbed. Each of the systems owns a subset of the APs at different locations, and the trajectories of the user are estimated by combining the inertial measurements from the foot-mounted IMUs and the WiFi beacons received from the APs.

(a) (b)

Figure 6.7: (a) The WiFi access points deployed for 4 different WiFi+IMU positioning systems ps_1, ps_2, ps_3 and ps_4 on the 4th floor testbed. APs shared by different systems are marked with the rectangles. (a) The WiFi access points deployed for 3 different WiFi+IMU positioning systems ps_5, ps_6 and ps_7 on the basement testbed. APs shared by different systems are marked with the rectangles.

4 days. The access point locations of different positioning systems are shown in Figure 6.7. During each day, we ran the experiments for approximately 3$\sim$4 hours, due to the limited battery life of the IMUs. The ground truth is collected by the users: the map of the floor is displayed on their mobile devices, and they tap the positions the are in to log their coordinates.

Pre-processing: The collected data is firstly cleaned to retrieve the meaningful trajectories (the timestamps that the users are actually moving) by thresholding the accelerometer readings, subsampled at a rate of 0.5 Hz. We assume space is discrete, i.e. it is a finite set $L = \{l_k\}$ with N discretized locations. In our experiment, the size of a discrete location is $3m \times 3m$. For the 4th floor testbed, the total number of locations $N = 209$ and for the basement testbed $N = 132$. The trajectories are then discretized according to the discre-



(a) (b)

Figure 6.8: (a) The 51 nodes used in our experiments, where the black blocks indicate the sensors whose data is used for test, while the circles indicate the sensors whose data is used as ground truth. (b) Node locations of the created sensor system sn_1 and sn_2. sn_1 has more nodes on the left part, while sn_2 dominates the right part. Sensors that are virtually "shared" by the two networks are grouped by rectangles.

tised set of locations L. For a given timestamp, the measurement from a positioning system is a probability vector of length N, where the j-th probability represents the belief of the system that the user is at location l_j.

6.1.2 Indoor environmental monitoring scenario

Data collection: The second experiment scenario considered in this thesis is an indoor environmental monitoring scenario. The data is collected from the widely used Intel Lab dataset [27], which contains temperature, humidity and light data received from 54 sensors (Mica2Dot sensors with weather boards) deployed in the Intel Berkeley Research lab for more than a month. The sensors reported data once every 31 seconds, and the data was forwarded by the TinyDB system [143] running on TinyOS. In our experiments, we consider the light intensity measurements, and select the readings of 5 consecutive days. We divide the 51 sensors (three sensors are omitted since they failed midway) into two groups randomly, where 26 of them are used to create the virtual sensor systems as explained below, and the rest of them are used as the ground truth to verify the proposed approaches. Figure 6.8(a) shows the locations of the nodes.

Sensor systems: Based on the collected dataset, we create two virtual sensor systems sn_1 and sn_2, which are coexisting and overlapping in space. sn_1 has 17 nodes, which are densely deployed in the left part of the environment. On the other hand, sn_2 has 18 nodes, where the majority of them are deployed on the right. The locations of the nodes belonging to each system are shown in Figure 6.8(b).

Pre-processing: The collected data is firstly processed as discussed in the pre-processing layer of the accuracy estimation framework (Section 3.2). In the original dataset, each

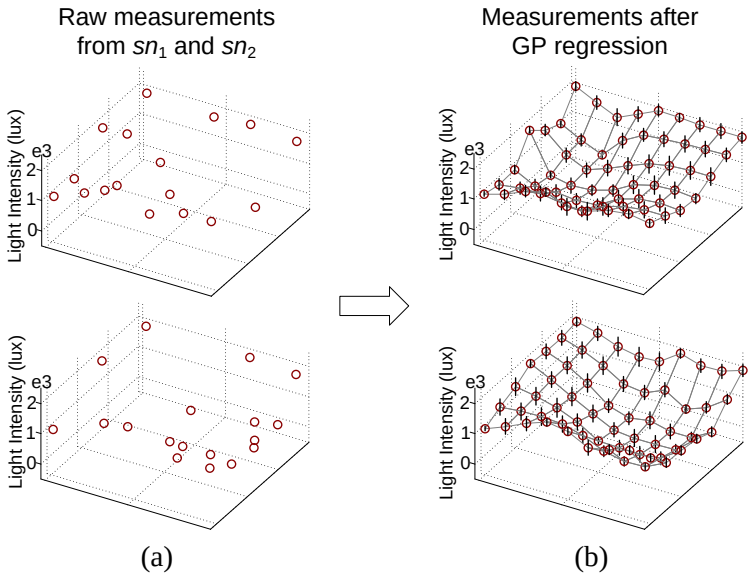


Figure 6.9: (a)The raw sensor measurements from systems sn_1 (top) and sn_2 (bottom), where the systems only report light intensity reading at the locations where they have nodes deployed. (b) The measurements generated by Gaussian process non-linear regression. Now measurements from systems sn_1 (top) and sn_2 have the same space granularity, and are converted into the probabilistic form.

node reports a light intensity measurement (Lux), which is a deterministic value between 0 to 100,000 every 31 seconds. The data from all sensors is synchronised to UNIX time, but the starting time of the nodes is different, and some sensors may fail to report data occasionally. At a given timestamp, in the raw dataset typically only readings from a subset of sensors are available. Therefore in our experiments we consider a re-sample step, which has two objectives: a) ensure the sensor measurements have the same time and space granularity; and b) convert the deterministic measurements to the probabilistic ones. For time granularity, we consider a linear interpolation approach. We interpolate the reported raw measurements from each sensor, and then subsample the interpolated data at a fixed interval of 5 minutes (we find that the light intensity dose not change very much within that amount of time).

Based on this data, we use the Gaussian process (GP) non-linear regression [53] to further interpolate over space. We perform this step for the two sensor systems sn_1 and sn_2 independently. At a given timestamp, the sensor systems (sn_1 and sn_2) only report light intensity readings at the locations where they have nodes deployed, as shown in Figure 6.9(a). Therefore the task is to use the reported readings as training points, and evaluate the light intensity values and the uncertainty associated with them over the entire space.

We assume that the light intensity values at different locations are correlated, and can

be modeled as a Gaussian process $\mathcal{GP}$. $\mathcal{GP}$ is defined as a distribution over the functions $f(\boldsymbol{x})$, which map an input vector $\boldsymbol{x}$ to a real value in $\mathbb{R}$. For a given $\boldsymbol{x}$, the function value $f(\boldsymbol{x})$ can be viewed as a random variable, and any finite subset of those variables follow a joint Gaussian distribution. In our case, we assume that light intensity is a function of the location, i.e. $\boldsymbol{x}$ is the 2D coordinates of the location, and $f(\boldsymbol{x})$ is the light intensity value of that location. Then the Gaussian process $\mathcal{GP}$ is defined in the 2D space, where at any given subset of locations the light intensity values are assumed to be normally distributed.

A Gaussian process can be fully determined by its mean function $m(\boldsymbol{x})$ and covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$, which are defined as:

$$m(\boldsymbol{x}) = E[f(\boldsymbol{x})]; \tag{6.2a}$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = E[(f(\boldsymbol{x} - m(\boldsymbol{x}))(f(\boldsymbol{x}' - m(\boldsymbol{x}'))] \tag{6.2b}$$

where the mean function describes the expected function values for the given input vector $\boldsymbol{x}$, i.e. the expected light intensity at a given location. The covariance function governs how the function values for two input vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ are correlated, i.e. the correlations between the light intensity values at two different locations. In our experiments, we use a simple mean function $m(\boldsymbol{x}) = c$, where c is a constant learned from the observed data. For the covariance function, we consider a summation of two radial basis function (RBF) kernel:

$$\begin{aligned} k(\boldsymbol{x}, \boldsymbol{x}') &= k_1(\boldsymbol{x}, \boldsymbol{x}') + k_2(\boldsymbol{x}, \boldsymbol{x}') + c \\ &= \sigma_1^2 \exp(-\frac{r^2}{2l_1^2}) + \sigma_2^2 \exp(-\frac{r^2}{2l_2^2}) + c \end{aligned} \tag{6.3}$$

where $r = (\boldsymbol{x} - \boldsymbol{x}')^{\mathrm{T}}(\boldsymbol{x} - \boldsymbol{x}')$, and in our case r is the squared Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x}'$: $r = \|\boldsymbol{x} - \boldsymbol{x}'\|^2$. σ_1^2, σ_2^2 are the scale factors, l_1, l_2 are the characteristic length-scale, and c is a constant. In our experiments, the two kernels k_1 and k_2 are used to model correlations between near and far locations, i.e. the parameters l_1 and l_2 are set differently. Given the dimension of the environment ($45m \times 35m$), we set $l_1 = 3$ and $l_2 = 9$. The other parameters σ_1^2, σ_1^2 and c are learned from the training data with the maximum likelihood approach discussed in [53].

Under the above setting, for each timestamp we run the Gaussian process non-linear regression for sensor systems sn_1 and sn_2, and an example of the results is shown in Figure 6.9(b). We can see that after this step, the measurements from the two sensor systems have the same space granularity. Also since the estimated values produced by this process are in the form of Gaussian, i.e. the estimated light intensity value at a give location is normally distributed, the deterministic measurements are now converted into probabilistic ones.

6.2 Competing Algorithms and Evaluation Metrics

In our experiments, we consider the following competing algorithms for accuracy estimation, note that they will all be evaluated on the pre-processed datasets:

Oracle Algorithm (OA): This algorithm has access to the real state x_t (i.e. the ground truth), and for a given measurement z_t^m, the accuracy it computes is the real accuracy, i.e. $f_{\epsilon_O}(z_t^m) = f_\epsilon(z_t^m; x_t)$. The computed accuracy of measurements can be indexed over given attributes or the monitored states, which is known in this case.

Report-based Algorithm (RA): This algorithm trusts the uncertainty reported by the sensor systems, and uses the mean $\bar{z}_t^m$ of the measurement as the estimated state to evaluate the accuracy, i.e. $f_{\epsilon_R}(z_t^m) = f_\epsilon(z_t^m; \bar{z}_t^m)$. The computed accuracy of measurements can be indexed over given attributes. If the accuracy needs to be indexed over the monitored states, this algorithm uses the $\bar{z}_t^m$ as the ground truth.

Voting-based Algorithm (VA): This algorithm uses the voting-based state estimation approach discussed in Section 3.3.1, and takes the measurements from all coexisting sensor systems at a given time into account. For each timestamp t, the algorithm evaluates the distribution of the estimated state $\hat{x}_{t_V}$ by cumulative voting, and uses it to evaluate the accuracy of a measurement $f_{\epsilon_V}(z_t^m) = f_\epsilon(z_t^m; \hat{x}_{t_V})$. The estimated accuracy can be indexed over given attributes, and if the accuracy needs to be indexed over the monitored states, this algorithm uses the mean of the distribution $p(\hat{x}_{t_V})$ as the ground truth.

Static Inference-based Algorithm (SIA): This algorithm uses the static inference-based approach discussed in Chapter 4, and considers all sensor measurements generated at timestamp t with known model parameters. Given the sensor measurements $z_t^{1:M}$ and prior ρ_t observed at t, it computes the estimated states $\hat{x}_{t_{SI}}$ as the posterior state estimate given the observed data: $p(\hat{x}_{t_{SI}}) = p(x_t | z_t^{1:M}, \rho_t)$. It then uses the estimated states to compute the accuracy $f_{\epsilon_{SI}}(z_t^m) = f_\epsilon(z_t^m; \hat{x}_{t_{SI}})$. The estimated accuracy can be indexed over given attributes, and if the accuracy needs to be indexed over the monitored states, this algorithms uses the mean of the distribution $p(\hat{x}_{t_{SI}})$ as the ground truth.

Static Learning-based Algorithm (SLA): This algorithm uses the static learning-based approach discussed in Chapter 5. It considers the sensor measurements $z_t^{1:M}$ and prior ρ_t at single timestamps t, and first learns the model parameters that are most consistent with the observed data. It then computes the estimated states $\hat{x}_{t_{SL}}$ with the learned parameters in the same way as *SIA* does, and uses the estimated states to evaluate accuracy $f_{\epsilon_{SL}}(z_t^m) = f_\epsilon(z_t^m; \hat{x}_{t_{SL}})$. The estimated accuracy of sensor measurements can be indexed over given attributes, and if the accuracy needs to be indexed over the monitored states, this algorithms uses the mean of the distribution $p(\hat{x}_{t_{SL}})$ as the ground truth.

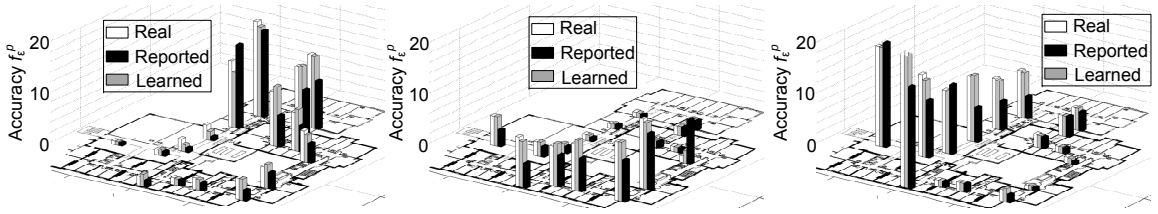
Dynamic Inference-based Algorithm (DIA): This algorithm uses the dynamic inference-based approach discussed in Chapter 4. It assumes monitored states are temporally correlated, and evaluates the estimated state sequence $\hat{x}_{1:T_{DI}}$ with all the observed measurements $z_{1:T}^{1:M}$ and priors $\rho_{1:T}$ under the known model parameters. The algorithm then computes the accuracy of a given measurement z_t^m with the t-th element in the estimated state sequence $\hat{x}_{1:T_{DI}}$ as $f_{\epsilon_{DI}}(z_t^m) = f_\epsilon(z_t^m; \hat{x}_{t_{DI}})$. The estimated accuracy of sensor measurements can be indexed over given attributes, and if the accuracy needs to be indexed over the monitored states, this algorithms uses the mean of the distribution $p(\hat{x}_{t_{DI}})$ as the ground truth.

Dynamic Learning-based Algorithm (DLA): This algorithm uses the dynamic learning-based approach discussed in Chapter 5. Similar to *DIA*, this algorithm also assumes the monitored state varies over time. But instead of relying on the model parameters known in advance, it firstly takes all measurements $z_{1:T}^{1:M}$ and priors $\rho_{1:T}$ into account and learns the parameters accordingly. If the accuracy needs to be indexed over the given attributes, the algorithm computes the state sequence $\hat{x}_{1:T_{DL}}$ with the learned parameters in the same way as in *DIA*, and evaluates measurement accuracy with respect to the estimated states: $f_{\epsilon_{DL}}(z_t^m) = f_\epsilon(z_t^m; \hat{x}_{t_{DL}})$. The evaluated accuracy is then aggregated and indexed over the given attributes. In the cases where the accuracy needs to be indexed over the monitored states, the algorithm uses the approaches discussed in Section 5.1.2, and computes the indexed accuracy directly from the learned model parameters.

We evaluate the above competing algorithms against the following metrics:

Accuracy Estimation Error EE^A. This metric describes the quality of the estimated accuracy with respect to the real accuracy. For a measurement z_t^m, the accuracy estimation error EE^A is defined as the squared difference between the estimated accuracy and the real accuracy (which is evaluated by the oracle algorithm *OA*): $EE^A(z_t^m) = (f_\epsilon(z_t^m) - f_{\epsilon_O}(z_t^m))^2$, where $f_\epsilon(z_t^m)$ is the accuracy estimated by any of the above competing algorithms. For aggregated accuracy I_ϵ^m which itself is the average over accuracy of multiple sensor measurements, $EE^A(I_\epsilon^m)$ is also defined as the squared difference between the estimated and the real accuracy: $EE^A(I_\epsilon^m) = (I_\epsilon^m - I_{\epsilon_O}^m)^2$, where $I_{\epsilon_O}^m$ is the aggregated accuracy evaluated by the oracle algorithm *OA*.

Running time RT. In all our experiments, the running time RT of an algorithm is the wall-clock time that the algorithm requires to finish its execution.



(a) (b) (c)

Figure 6.10: The real, reported and learned accuracy (computed by *DLA*) for positioning system ps_1 (left), ps_2 (middle) and ps_3 (right).

6.3 Experiment Results

We have implemented all the algorithms in Section 6.2 in MATLAB 8.0, and all the experiments were performed on a quad-core machine with Linux 2.6.32. In our experiments, we use the proximity-based accuracy metric f_ϵ^p for the indoor positioning scenario (introduced in Section 6.1.1), and the similarity for environmental monitoring scenario (introduced in Section 6.1.2).

6.3.1 Accuracy of sensor systems varies over time and space

The first set of experiments shows that the accuracy of a sensor system can vary over time and space, while the reported accuracy may not be a good indicator of the real accuracy. Note that here we temporarily assume that the ground truth of the monitored states is known in all experiments within this subsection.

Positioning scenario: Figure 6.10 shows that the real accuracy (averaged over the measurements received in all timestamps) of the coexisting positioning systems ps_1, ps_2 and ps_3 (the white bars) vary significantly over space. Referring to Figure 6.7, which shows the locations of the deployed sensors of $ps_1 \sim ps_3$, we can see that the accuracy of a positioning system is higher in areas where it has denser sensing infrastructure. In this experiment we see that ps_1 has good accuracy (shorter white bars) at the left bottom part of the floor (Figure 6.10(a)), while ps_2 performs well on the right side (Figure 6.10(b)), and ps_3 dominates the top area ((Figure 6.10(c))). The experiments also show that the reported accuracy is not always reliable: the reported accuracy (black bars) consistently over or under estimates the real accuracy (white bars). The accuracy computed by the proposed dynamic learning algorithm (*DLA*) (grey bars) is much closer to the real accuracy (white bars). This shows that in the absence of ground truth, the real accuracy can be effectively approximated by applying the proposed techniques.

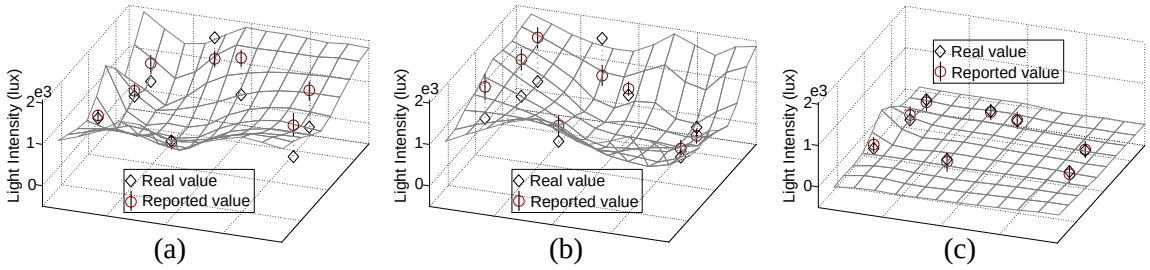


Figure 6.11: 3D snapshots showing that the real accuracy varies over space and time. The surfaces show the light intensity measurements (only the means) across space at different timestamps. The first two graphs show real and reported light intensity data generated at daytime by sensor networks sn_1 (left) and sn_2 (middle). The right graph shows real and reported light intensity data generated by sn_1 at night.

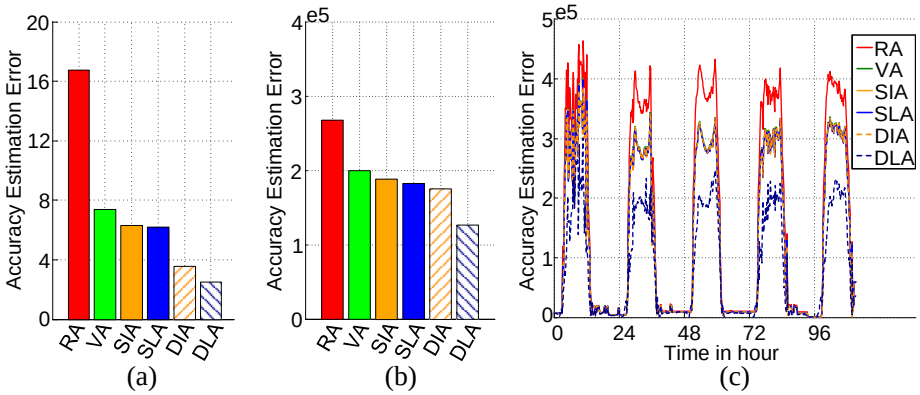


Figure 6.12: (a) Average accuracy estimation errors of different approaches in the indoor positioning scenario. (b) Average accuracy estimation errors of different approaches in the indoor environmental monitoring scenario. (c) Average accuracy estimation errors of different approaches in the indoor environmental monitoring scenario vary over time.

Environmental monitoring scenario: For the indoor environmental monitoring scenario, Figure 6.11 shows that the real accuracy of a sensor system can vary over both location and time. Figures 6.11(a) and 6.11(b) show snapshots of the light measurements reported at daytime by systems sn_1 and sn_2 respectively. We can see that: a) the differences between the real light values and reported ones vary across space, and b) the reported accuracy (variance) is very unreliable and the real light values consistently fall out of the 95% confidence intervals of the reported ones. Figure 6.11(c) shows a snapshot of real and reported values (by sn_1) at night; notice that the differences here between real and reported values are very small, which suggests that sn_1 becomes accurate everywhere at night.

6.3.2 Accuracy estimation performance

In this set of experiments, we compare the performance of the accuracy estimation algorithms in terms of *Accuracy Estimation Error* (EE^A), averaged over all measurements.

Positioning scenario: Figure 6.12(a) shows the accuracy estimation errors of different algorithms (*RA*, *VA*, *SIA*, *SLA*, *DIA*, *DLA*) in the indoor positioning scenario when we consider all the probabilistic measurements from coexisting positioning systems ps_1, ps_2 and ps_3. We can see that in this case the voting-based algorithm (*VA*) works very well, and can reduce more than half of the estimation errors of the report-based algorithm. This shows that the measurements from coexisting sensor systems can indeed help to improve the estimation of accuracy, and simple approaches like voting could be quite effective in practice. Techniques that only operate on single timestamps (*SIA* and *SLA*) can provide about 10% reduction of estimation errors compared to voting, since in our experiments we only have a limited number of information sources (three in this case). The improvement actually comes from the model parameters, which determine the weights of measurements from different sensor systems when combining them. Note that the difference in estimation errors of static inference and learning (*SIA* and *SLA*) is negligible, because at a single timestamp we only have a few measurements, and it is not possible to improve the model parameters significantly.

The dynamic algorithms (*DIA* and *DLA*) are generally better than their static counterparts. Dynamic inference (*DIA*) features more than 40% improvement compared to voting (*VA*), because it takes all measurements into account and uses a state transition model that reflects the underlying state dynamics. Dynamic learning (*DLA*) can further reduce the estimation error to less than half of voting (*VA*), since *DLA* also learns the model parameters that best explain the stochastic measurements. We also see that compared to dynamic inference (*DIA*), dynamic learning (*DLA*) offers about 20% benefit due to the extra learning process. Finally, in this scenario the improvement from the naive report-based algorithm (*RA*) to the best technique (*DLA*) is almost eight fold.

Environmental monitoring scenario: For the indoor environmental monitoring scenario, as shown in Figure 6.12(b), there is a similar trend of improvement as we move to more sophisticated techniques. Comparing with the naive approach *RA*, voting (*VA*) in this case can provide approximately 30% improvement in estimation errors. However, the improvement from voting to static inference *SIA* is negligible. This is because the measurement model (the H_t^m discussed in Chapter 4) is derived directly from the reported confidence intervals of the reported data, which often do not cover the ground truth (as shown in Figure 6.11). The difference in estimation errors between static inference and learning (*SIA* and *SLA*) is also marginal, due to the limited number of measurements observed within one timestamp.

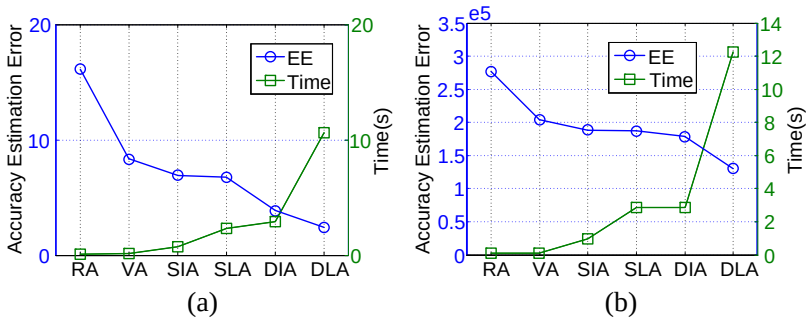


Figure 6.13: (a) Running time vs. performance for different algorithms in the indoor positioning scenario. (b) Running time vs. performance for different algorithms in the indoor environmental monitoring scenario.

In this scenario dynamic inference (DIA) fails to provide significant improvement, again because the initial estimate of the model parameters is poor. The benefits of dynamic learning *DLA*, however, are far more pronounced, since the learned model parameters are more accurate, and can explain the observed data better.

Figure 6.12(c) shows that in this scenario the relative performance of different algorithms varies significantly over time. We can see that during the day time when the light intensity is high, the accuracy estimation errors of the approaches clearly fall in three groups: the report-based algorithm *RA* is high up, the voting- and inference-based approaches (*VA*, *SIA* and *DIA*), together with static learning *SLA* is about 30% better than *RA*, while dynamic learning *DLA* can approximately halve the estimation error of *RA*. However, during the night when the light intensity is low everywhere, all of the algorithms perform similarly.

6.3.3 Running cost vs. performance gain

In this set of experiments, we study the trade-off between accuracy estimation and the computation cost of different approaches. We measure the execution time of different algorithms and compare it with the performance gain in terms of accuracy estimation error EE^A. Figure. 6.13 shows the trade-off in the indoor positioning and environmental monitoring scenarios. We observe similar trends in both scenarios. Firstly, voting is very efficient, since it offers a significant performance gain (more than half in the indoor positioning scenario and about 30% improvement in the indoor environmental monitoring scenario) with very little additional running cost. This is because it combines the measurements observed at one timestamp with equal weights to estimate the latent states, which turns to work quite well in our experiments. Static inference *SIA* is slightly more expensive than voting, since it uses the known model parameters to weight different measurements before combining them (as discussed in Chapter 4). Static learning *SLA* is 3-4 times slower than static inference

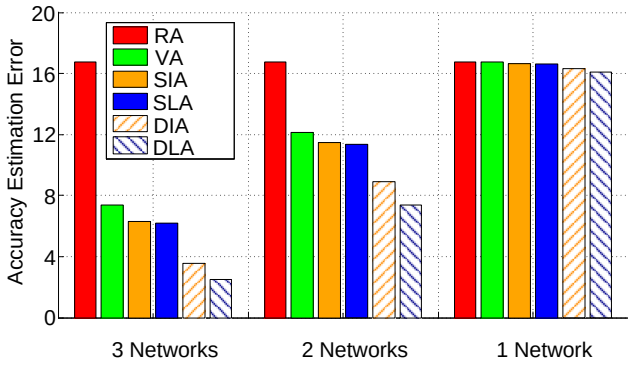


Figure 6.14: In the indoor positioning scenario, the average accuracy estimation errors of different approaches when the number of coexisting positioning systems varies from 3 to 1.

SIA. In fact, generally learning-based techniques are more expensive than inference-based, since learning requires iterative evaluation of the likelihood of data through multiple runs of inference. However the performance gain from static inference *SIA* to static learning *SLA* is marginal in both scenarios, as shown in the previous set of experiments. Dynamic approaches are naturally more expensive, because they use all the observed measurements to evaluate the sequence of latent states. In the indoor positioning scenario, dynamic inference *DIA* offers about 50% improvement but only requires marginally more running time compared to static learning *SLA*. It does not improve much in the environmental monitoring scenario since the initial model parameters are not good enough, as discussed in the previous set of experiments. The dynamic learning algorithm *DLA* is the most expensive one, but also the best algorithm. In both scenarios, when moving from dynamic inference *DIA* to dynamic learning *DLA*, the performance gain is about 30%$\sim$40% at the expense of about 3$\sim$4 fold increase in running time.

6.3.4 Sensitivity to the number of information sources

In this experiment, we study the sensitivity of different approaches to the number of information sources, i.e. the number of coexisting sensor systems. We only consider the indoor positioning scenarios, since the environmental monitoring scenario only have two coexisting sensor systems. Figure 6.14 shows the average accuracy estimation errors of different approaches when the number of coexisting positioning systems varies from 3 to 1.

We can see that with fewer coexisting systems, the performance gaps between the different techniques become smaller. In the case where two systems are available, we see that instead of halving the estimation errors as shown in Section 6.3.2, now voting (*VA*) can only provide about 25% of improvement. Similarly, the gap between static approaches

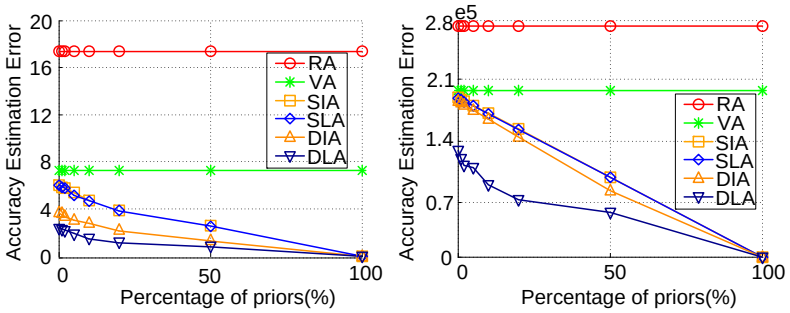


Figure 6.15: (a) In the indoor positioning scenario, the accuracy estimation errors of different approaches when the percentage of priors varies. (b) In the indoor environmental monitoring scenario, the accuracy estimation errors of different approaches when the percentage of priors varies.

(*SIA* and *SLA*) and voting is also smaller (less than 10%). In this case, dynamic approaches can still improve, where dynamic inference (*DIA*) yields 40% less estimation error than the naive report-based algorithm, and dynamic learning (*DLA*) can halve the estimation errors. In the case where only one network is available, the best performing algorithm (*DLA*) has similar estimation error to the naive approach of trusting the reported accuracy (*RA*), since there is no other information available to infer the actual accuracy of the measurements.

6.3.5 Sensitivity to the priors

In this set of experiments, we show how the prior knowledge on the monitored states can influence different accuracy estimation approaches. For both indoor positioning and environmental monitoring scenarios, the priors are generated by first selecting a random subset of the timestamps. At these timestamps, the prior distribution $p(\rho_t)$ is set to be the ground truth value plus a small quantity of noise. We vary the percentage of timestamps that have priors, and study the effect on the performance of the competing accuracy estimation algorithms.

Figure 6.15(a) shows that in the indoor positioning scenario, as the percentage of priors increases, the static approaches (*SIA* and *SLA*) improve linearly. For dynamic approaches (*DIA* and *DLA*), the estimation errors have a quick drop before the percentage of priors reaches 20%, and then become flat. This is because the dynamic approaches exploit temporal correlations in the data, which enables prior knowledge to impact previous and future states. Note that there is also a gap between dynamic inference and learning, which first increases until the prior reaches 10%, and then decreases afterwards. This is because the dynamic learning *DLA* can incorporate the priors during each learning iteration, which re-

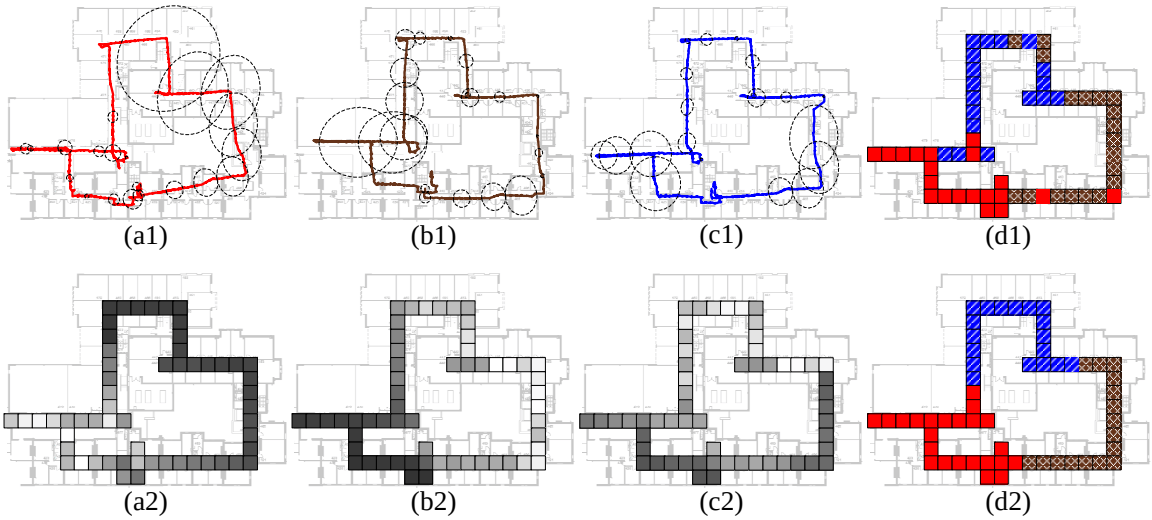


Figure 6.16: (a1) Estimated trajectory from positioning system ps_1. (a2) Real accuracy of ps_1 (lighter means more accurate) (b1) Estimated trajectory from ps_2. (b2) Real accuracy of ps_2 (lighter means more accurate) (c1) Estimated trajectory from ps_3. (c2) Real accuracy of ps_3 (lighter means more accurate) (d1) Each traversed location is labeled with the locally most accurate positioning system. (d2) Space is clustered into regions where each of them is dominated by a positioning system.

inforces the prior information in the model parameters and thus biases the estimated states. In the indoor environmental monitoring scenario, a similar behavior can be witnessed, as shown in Figure 6.15(b). However, note that the gap between dynamic inference *DIA* and learning *DLA* is larger. This is because in this case, the model parameters used by dynamic inference *DIA* are inaccurate, while *DLA* can learn them from the observed data, which further clamps down the estimation errors.

6.3.6 Ranking sensor systems based on accuracy

In this set of experiments, we show that in practice we can make informed decisions as to which sensor systems to task based on their accuracy information. We consider the indoor positioning scenario, where three positioning systems ps_1, ps_2 and ps_3 are tracking the users, and the task is to decide which positioning system to use at different locations.

Figures 6.16(a1), (b1) and (c1) show the reported trajectories and the reported accuracy (the error ellipses) when each of the three systems is tracking a user along the corridor. Figures 6.16(a2), (b2) and (c2) show the real accuracy of the three positioning systems at different locations, which is evaluated by the oracle algorithm *OA* with respect to the ground truth. It is obvious that none of the systems is globally better than the other. Recall from Figure 6.7 that ps_1 has most of its nodes at the bottom left corner, ps_2 dominates on

the right side, and ps_3 on the top side of the floor, one can see that each of the systems tends to excel in the area where it has the denser infrastructure.

Based on the real accuracy, Figure 6.16 (d1) shows the most accurate positioning system at each location visited by the user. We see that generally on the bottom left part of the floor, ps_1 should be used, while ps_2 should be tasked when the user is on the right side, and ps_3 should be used on the top. This allows us to actively switch between the positioning systems in search for the most accurate systems. To avoid switching too often, we transform Figure 6.16(d1) to Figure 6.16(d2), by applying a k-means based clustering algorithm which finds broader regions dominated by a positioning system. With the space partitioning shown in Figure 6.16(d2), the user can decide which positioning system to use where.

6.3.7 Ranking performance

The previous set of experiments have shown that, knowing the accuracy of the sensor systems is vital for the users to select the best system to task in different contexts. In this set of experiments, we show that the accuracy of different algorithms can affect the user's ability to rank the sensor systems, and to choose the best one to use. We also consider the positioning scenario, and assume that the user would like to use the most accurate positioning systems as she moves to different locations. Therefore in this case, the ranking of the positioning systems is determined based on their accuracy indexed over the locations of the user, i.e. the monitored states.

We only consider the following algorithms: the oracle *OA*, the report-based *RA*, the voting-based *VA* and the dynamic learning-based (*DLA*) algorithms, since previous experiments have shown that dynamic learning *DLA* is superior to static inference *SIA*, static learning *SLA* and dynamic inference *DIA*. Note that in this case dynamic learning *DLA* uses the technique discussed in Section 5.1.2, which directly estimates the accuracy of the positioning systems at different locations from the learned emission probabilities.

We examine four different scenarios, the results of which are depicted in the four rows of Figure 6.17 respectively.

Scenario 1: The first scenario (row 1 of Figure 6.17) includes positioning systems ps_1, ps_2 and ps_3 deployed on the 4th floor testbed and a user traversing it holding a Nexus S device. Observe that the proposed dynamic learning algorithm *DLA* (Figure 6.17(a4)) manages to partition the space in almost the same way as the oracle algorithm (*OA*) (Figure 6.17(a1)) that has perfect knowledge of the accuracy of the systems. In this scenario, the two competing algorithms (*RA* and *VA*) perform reasonably well (Figure 6.17(a2) and Figure 6.17),

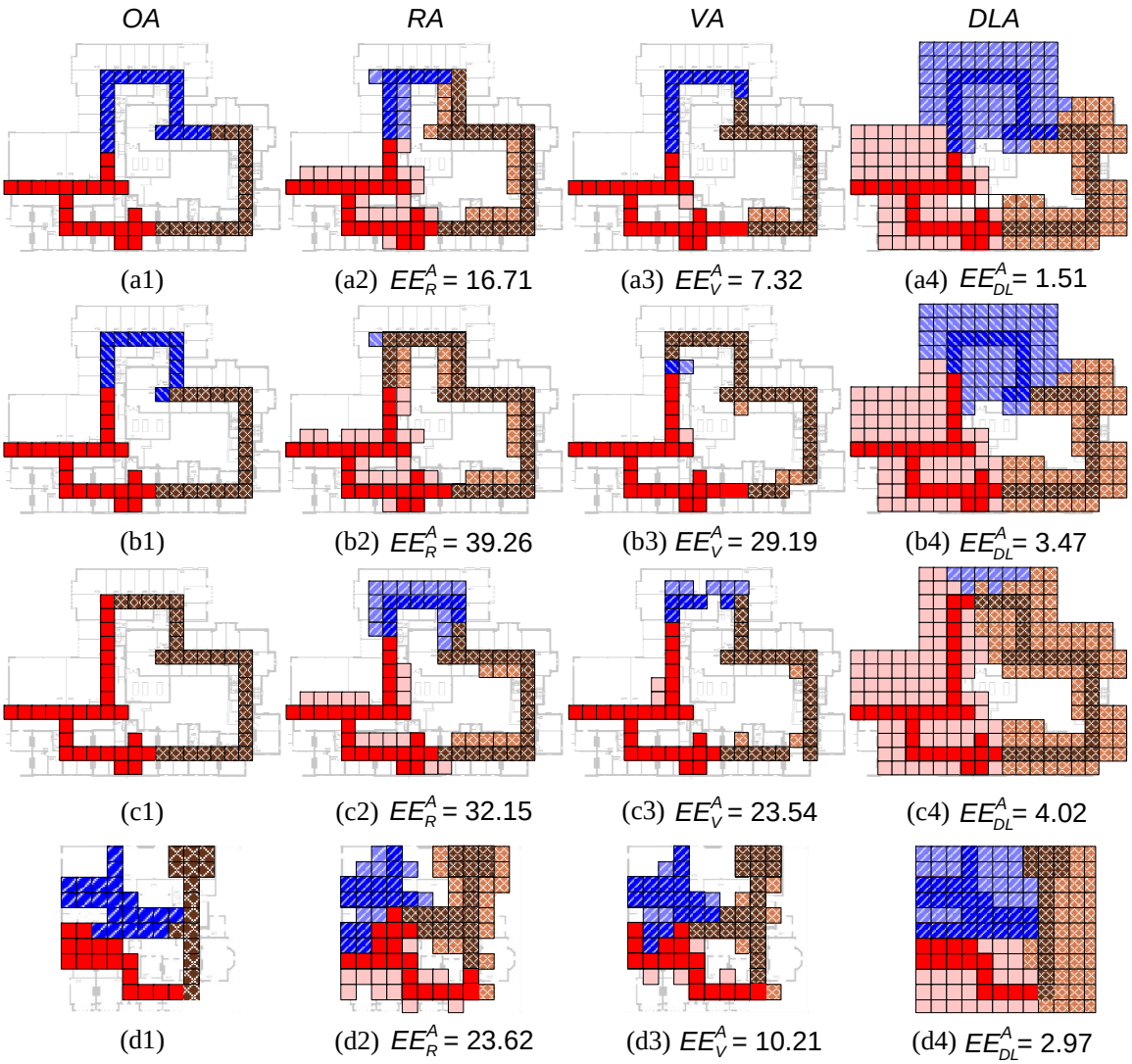


Figure 6.17: Maps showing preferences of the positioning systems and the accuracy estimation errors of different algorithms: *OA*, *RA*, *VA* and *DLA*, with the trajectory locations highlighted and empty blocks indicate no information there. Row 1: Testbed = 4F, User = Nexus S, Systems = $\{ ps_1, ps_2, ps_3\}$; Row 2: Testbed = 4F, User = Nexus S, Systems = $\{ ps_1, ps_2, ps_4\}$; Row 3: Testbed = 4F, User = TF201, Systems = $\{ ps_1, ps_2, ps_3\}$; Row 4: Testbed = 0F, User = Nexus S, Systems = $\{ ps_5, ps_6, ps_7\}$. In all scenarios we assume that in 10% of the timestamps we have prior information on user locations, which is used by *DLA*.

with the report-based algorithm *RA* making a few errors in the upper part of the user's trajectory.

Scenario 2: In the second scenario (row 2 of Figure 6.17), we use a different set of co-existing positioning systems: ps_1, ps_2 and ps_4, where ps_4 has the same infrastructure as ps_3 but more conservative noise profile (with twice as high gyroscope and accelerometer variances). Although this does not change the ground truth much, it throws off the two competing algorithms *RA* and *VA*, which now refrain from using ps_4 on the top part of the trajectory. Notice however that the proposed dynamic learning-based algorithm *DLA* still perceives ps_4 to be the best there and still matches the *real* partitioning.

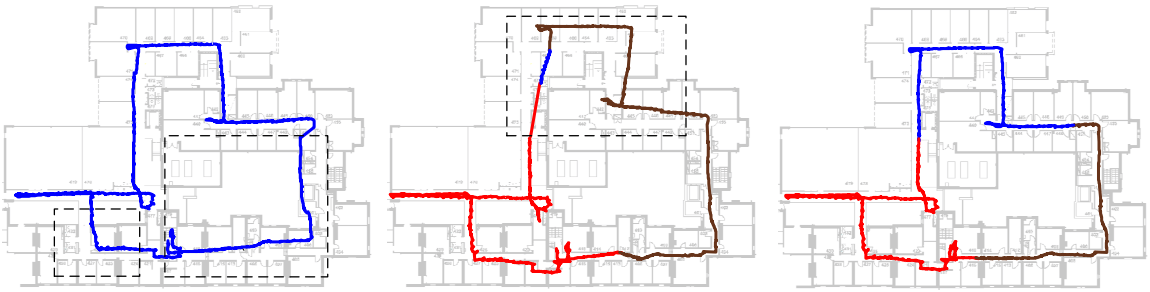
Scenario 3: In the third scenario, (row 3 of Figure 6.17) we use the same setup as in the first scenario (row 1), except that the user now carries an Asus TF201 tablet instead of the Nexus S phone. Whereas ps_1 and ps_2 use different radio propagation models for the two devices, ps_3 has been tuned for phone users only. Hence, it performs poorly on the tablet and ceases to perform well on the top part of the user's trajectory (see Figure 6.17(c1)). Notice that only the proposed dynamic learning algorithm *DLA* manages to detect the drop in ps_3's accuracy and partition the space correctly, whereas the other two algorithms *RA* and *VA* still prefer ps_3 on the upper part of the floor.

Scenario 4: In the fourth scenario (row 4 of Figure 6.17), we change the venue to the basement testbed, which has more open areas than the 4th floor testbed. Here, the competing algorithms *RA* and *VA* erroneously show a preference on using ps_6 not only on the right hand side of the floor, but also further into the center (Figure 6.17(d2) and 6.17(d3)). The proposed dynamic learning algorithm *DLA* (Figure 6.17(d4)) again partitions the space as if it had access to the real accuracy of the positioning systems (Figure 6.17(d1)).

6.3.8 Impact of the ranking performance

The final set of experiments are set up to investigate how ranking performance of different algorithms has a knock-on effect on using the coexisting sensor systems. We consider the same settings as in the previous set of experiments, if a user relies on the space partitioning computed by the proposed learning-based algorithm *DLA*, to switch from one positioning system to another, will they be localised more accurately, than if they used that of the competing algorithms?

Here we compare only with the voting-based algorithm *VA*, since it has proved to be superior to the naive report-based algorithm *RA* in the previous experiments. We also compare with the strawman approach of using one positioning system everywhere, i.e. the one that has the best overall accuracy. In our experiment, as the user moves across the floor, we switch positioning systems according to the space partitioning generated by



(a) (b) (c)

Figure 6.18: (a) Trajectory generated by the single best positioning system. Dashed rectangles indicate the areas where the error of measurements (evaluated by f_ϵ^p with respect to the ground truth) are larger than certain threshold (4 in this case). (b) Switching according to the voting-based algorithm *VA*, and the average error of measurements (evaluated by f_ϵ^p with respect to the ground truth) is 2.30. (c) Switching according to the proposed dynamic learning-based algorithm *DLA*, and the average measurement accuracy (evaluated by f_ϵ^p with respect to the ground truth) is 1.44.

different algorithms (the strawman approach, *VA*, and *DLA*). We then evaluate the average accuracy of the position measurements produced by different switching plans respectively. Figure 6.18 shows that switching between different positioning systems is generally better than using the same system everywhere (comparing Figure 6.18(a) with Figure 6.18(b) and (c)). It also shows that relying on the space partitioning generated by the proposed learning-based algorithm *DLA* can reduce the errors in measurements (evaluated by f_ϵ^p with respect to the ground truth) by about 40% (2.30 to 1.44), when compared to that of the best competing algorithm *VA*.

6.4 Discussion

We have shown in this chapter the systematic experimental evaluation of the proposed accuracy estimation approaches, conducted in two real-world sensing scenarios. We have built two indoor positioning testbeds, and implemented an array of different indoor positioning systems as research vehicles for the accuracy estimation problem studied in this thesis. We ran the systems for more than 20 days, and have collected a rich set of positioning data. We also include the widely used Intel lab data as another dataset in our experiments, and show how to pre-process both datasets before estimating the accuracy. Most importantly, we have shown that the accuracy estimation approaches proposed in this thesis outperform the competing ones, and we have also compared the different approaches thoroughly. Our

key findings from the empirical study are as follows:

- The accuracy of sensor systems can vary significantly in different contexts, e.g. time and space, and in practice it is difficult to find one system that is superior overall. On the other hand, the reported accuracy is not always a faithful indicator of the real accuracy.

- The accuracy estimation performance of different approaches is very different, and can depend on various factors, such as the quality of reported data, the number of information sources, the amount of prior knowledge, etc. The cost of the approaches in terms of running time can also vary significantly.

- In static case where only the data within single timestamps is considered, inference- and learning-based approaches *SIA* and *DIA* are only marginally better than the voting-based approach *VA*. Therefore voting is preferred among the three due to its simplicity and low cost.

- Dynamic inference- and learning-based approaches *DIA* and *DLA*, however, can significantly reduce the accuracy estimation error when compared to voting (by more than 50% in some scenarios), since the correlations between the data in different timestamps are exploited.

- The merits of learning-based approaches *SLA* and *DLA* vs. the inference-based approaches *SIA* and *DIA* are more pronounced in the scenarios where we have a poor prior knowledge of the model parameters. When this is not the case, the inference-based approaches are preferred because their accuracy estimation performance is close to that of learning, but at a much lower computation cost.

- The more the coexisting sensor systems, the greater the relative benefits of voting-based approach *VA* compared to trusting reported accuracy (computed by *RA*) since more information is available. Similarly, the performance gap between the proposed inference / learning- based approaches and voting is larger with more coexisting sensor systems.

- Prior knowledge on the state distribution can significantly impact the relative performance of the accuracy estimation approaches. The more the timestamps on which we have priors, the smaller the accuracy estimation error of the inference- and learning-based approaches. Whereas static approaches *SIA* and *SLA* improve their performance linearly, dynamic approaches *DIA* and *DLA* improve faster with fewer priors because they exploit underlying dynamics to propagate the priors to nearby states.

- Based on the accuracy of the sensor systems, the users are able to make informed decisions as to which systems to task in different contexts. However, the tasking plan generated by different algorithms can be very different, and the proposed dynamic learning-based approach *DLA* outperforms the competing ones significantly: it can produce the tasking plans as if it has access to the ground truth.

- Different tasking plans of the sensor systems have a significant knock-on effect on the overall performance when using the sensor systems. Generally, actively switching to different systems is preferred than only using the single best systems, and the measurement error of switching systems according to the proposed dynamic learning-based approach *DLA* can be up to 40% less than that of the best competing approach *VA*.

However, the experiments explained in this chapter also have some limitations. Firstly, the accuracy patterns of the systems considered in our experiments are limited, e.g. the accuracy only varies over time, space or different users. Secondly, our experiments only considered the two accuracy metrics (the proximity- and similarity-based) proposed in this thesis. Thirdly, in our indoor positioning experiments, the ground truth is logged by the users manually (the map of the testbed is displayed on the screen of the devices carried by the users, and they touch the locations where they are to log the ground truth), which requires a lot of labour and may not be practical in other settings. Finally, the prior knowledge considered in our evaluation is restricted to the state distributions at single timestamps, where more complex types of prior knowledge, such as the ones that can span over multiple timestamps are not incorporated. In the next chapter, we will conclude this thesis, and highlights the potential areas for future work, which aim to address the above mentioned limitations.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis has studied the emerging problem of estimating the accuracy of coexisting sensor systems. We have described the motivation and challenges of the problem of accuracy estimation, and formulated it in a general way which covers a broad spectrum of sensing scenarios. We have proposed a general accuracy estimation framework, which breaks the problem of estimating accuracy down to layers and addresses it step by step. We have shown that the proposed accuracy estimation framework can be used in various cases, and implemented in different ways. Following the framework, we have proposed two distinct classes of approaches, the inference-based and the learning-based, to tackle the problem of accuracy estimation. The inference-based approaches firstly infer the monitored states, with respect to which the accuracy of sensor measurements is evaluated, and further aggregated to build accuracy indices of the sensor systems. On the other hand, the learning-based approaches incorporate parameter learning techniques in estimating accuracy, where the learned parameters are either used to improve the accuracy of state estimates, or to derive the indexed accuracy directly in certain cases. Finally, we have evaluated and compared an array of accuracy estimation approaches, including the proposed and competing approaches, in two real-world sensing scenarios, and shown that the proposed approaches outperform the competing ones significantly. Concretely, the novel contributions of this thesis are as follows:

Formulation of the accuracy estimation problem: We have motivated the problem of accuracy estimation from various real-world sensing scenarios, such as multiple sensing services competing for the same group of users, detecting faults in large scale networks, or establishing trustworthiness of different individuals in social sensing, etc. We have also described the key challenges arising in this accuracy estimation problem with illustrative

examples, for instance, the ground truth of the monitored states is absent, the users often lack a clear view of the implementation details of the sensor systems, and the reported accuracy can be misleading, etc. We have explained the important concepts and assumptions, and formulated the accuracy estimation problem in the sensor network context. We assume that the sensing application would like to monitor the physical phenomenon with certain accuracy requirements in mind, and multiple coexisting sensor systems offer services by providing probabilistic sensor measurements, which are probability distributions over the domain of monitored signals. We have also introduced the idea of prior knowledge on states, which can be acquired from a third party, e.g. the users, and provide clues on how the monitored signals are distributed. The goal is then to estimate the accuracy of the coexisting sensor systems according to the requirements of the application, given all the available sensor measurements and prior knowledge.

Accuracy estimation framework: We have proposed a novel accuracy estimation framework (in Chapter 3), which provides a general solution to the accuracy estimation problem, and can be used in a wide range of sensing scenarios. The proposed framework lies between the sensing application and the underlying sensors systems, and includes four layers: *pre-processing*, *state estimation*, *accuracy estimation* and *accuracy indexing*. The pre-processing layer retrieves the raw sensor measurements, synchronises them to the same clock, metric system and probabilistic form, and then resamples the measurements from different systems so they have the same time and space granularity. The state estimation layer then infers the monitored states given the pre-processed sensor measurements and available prior knowledge. We have shown that this layer can be implemented in a number of different ways, and created a taxonomy of approaches ranging from a simple voting scheme to more sophisticated inference and learning techniques. We have also shown that inference and learning can be further divided into static and dynamic variants, depending on whether the temporal correlations between the states are considered. With respect to the estimated states, the accuracy estimation layer evaluates the accuracy of sensor measurements according to an application-defined accuracy metric. We have introduced two different accuracy metrics, the proximity-based and similarity-based, and shown that they can be used by sensing applications with different types of accuracy requirements. Finally, the accuracy indexing layer aggregates the evaluated measurement accuracy over attributes specified by the sensing application, and builds accuracy indices of the coexisting systems accordingly.

Inference-based accuracy estimation approach: We have proposed a novel inference-based approach for accuracy estimation (in Chapter 4), which implements the proposed accuracy estimation framework. We have demonstrated that the existing probabilistic graph-

ical models, such as the hidden Markov models, are not appropriate in this case, since they only have one deterministic observation at a time, and do not consider prior knowledge on states other than at the beginning. We have proposed augmented models to capture the key elements in our context, which are able to emit multiple probabilistic measurements at a time, incorporate prior knowledge on states at any timestamp, and have model parameters with new semantics. Based on the augmented models, we have proposed new state inference algorithms, which are able to infer the latent states in the presence of multiple sequences of probabilistic sensor measurements and arbitrary amount of priors. For probabilistic measurements, we have shown that the proposed algorithms can take them into account by marginalisation. For multiple sequences of measurements, we have shown that the algorithms can incorporate them by assuming conditional independence between measurements generated by different systems. For priors on states, we have shown that the algorithms can bias state estimation by multiplying the prior state distribution before any measurements are considered. We have demonstrated that the proposed state inference algorithms work in both discrete and continuous cases, and explored two variants of the algorithms, the static and dynamic, depending on whether the temporal correlations of the states are considered. The static inference algorithms assume the states at different timestamps are independent, and estimate the states based on the measurements and priors observed within single timestamps, while the dynamic counterparts consider the states as a stochastic process, and use the sequences of measurements and priors to infer the states. Finally, we have illustrated that the quality of the estimated states can have substantial impact on the quality of accuracy estimation.

Learning-based accuracy estimation approach: We have also proposed a novel learning-based approach for accuracy estimation (in Chapter 5). We have shown that the proposed learning approach works in two different modes, depending on the accuracy requirements of the sensing application. When accuracy needs to be indexed over attributes other than the monitored state, we have shown that the proposed learning approach can implement the four-layer accuracy estimation framework, but employ parameter learning techniques to improve accuracy estimation, rather than rely on the model parameters known a priori. On the other hand, when the accuracy needs to be indexed over the monitored states, the proposed learning approach is able to skip the state estimation layer, merge the accuracy estimation and indexing layers of the proposed framework, and derive the aggregated accuracy from the learned model parameters directly. We have shown that for both variants of the proposed approach, estimating the model parameters that are the most consistent with the observed data is the key step, we have extended the existing parameter learning algorithms (e.g. the Baum-Welch algorithm for hidden Markov models) to enable them to

work on the augmented models used in our context. To address this, we have proposed new parameter learning algorithms based on the Expectation Maximisaion (EM) scheme, which are able to re-estimate the parameters given the multiple sequences of probabilistic sensor measurements and priors. The proposed algorithms have both static and dynamic variants, depending on how the observed data is used to re-estimate the parameters: static algorithms operate on slices of the observed data, while dynamic algorithms learn the parameters with full sequences of measurements and priors. We have shown that for both the discrete and continuous cases, the parameter learning problem can be formulated in such a way that is amenable to the general EM scheme, and solved analytically with the EM framework. We have also discussed several important implementation issues, and provided methods to prevent the problem of insufficient training data or over-fitting during the learning process.

Comprehensive experimental evaluation: Finally, we have conducted a systematic experimental evaluation of the proposed accuracy estimation approaches, on data collected from two real-world sensing scenarios: an indoor positioning scenario and an indoor environmental monitoring scenario. For the positioning scenario, we have built two testbeds in our department building, and implemented different classes of indoor positioning systems. We have tracked two users at the two testbeds for more than 20 days and collected a rich set of data. For the environmental monitoring scenario, we have collected the data from the Intel lab dataset [27], and shown that by applying Gaussian process (GP) non-linear regression, the raw sensor measurements can be interpolated to have the same time and space granularity, and converted to the appropriate probabilistic form. We have considered an array of different accuracy estimation approaches, including the naive report-based, the voting-based, and the proposed inference- / learning-based approaches. We have compared those approaches thoroughly on the datasets collected from both scenarios, and the key findings learned from the experiments are: 1) the accuracy of sensor systems can vary significantly in different contexts, and typically no system can be superior overall; 2) the accuracy estimation performance and running time of different approaches can be very different; 3) in our case where only a few sensor systems are available, static inference and learning are only marginally better than voting, but much more expensive; 4) dynamic inference and learning are significantly better than voting since the correlations between monitored states are considered; 5) learning is preferred to inference only if the initial knowledge on model parameters is poor, otherwise, the performance gap between them is not significant; 6) the more the coexisting sensor systems, the larger the gaps between the proposed inference / learning approaches and the simple approaches; 7) prior knowledge on states can significantly impact the performance of different approaches; 8) based on the

estimated accuracy, one can make informed decisions as to which systems to task in different contexts, and the proposed learning-based approach can generate high quality tasking plans; 9) different tasking plans have a significant impact on the overall performance of using the sensor systems, and switching systems according to plans produced by the proposed approach can improve performance significantly.

7.2 Future Work

This thesis also has some limitations. For instance, we have only considered the directed graphical models and recursive Bayesian techniques to capture the correlations between the monitored process and sensor observations. Another limitation is that the proposed approaches rely on the reported sensor measurements from different systems to validate each other, and could perform badly if all the systems are reporting erroneous measurements. On the other hand, the prior knowledge which can bias the states towards ground truth is limited: we have only considered the prior state distributions at single timestamps. Finally, the experiments in this thesis have only considered systems whose accuracy patterns are relatively simple and do not change rapidly, while in practice the accuracy of sensor systems could be more complex. Therefore, the possible directions of future work that may address the above limitations include:

Employ undirected graphical models for accuracy estimation: The first direction of future work we would like to explore is to use undirected graphical models to address the accuracy estimation problem. For example one possible approach is to use conditional random fields (CRFs), which were firstly introduced in [144], and have been widely used in areas of natural language processing, image processing and bioinformatics. Unlike from the directed models used in this thesis, CRFs do not have to explicitly model the probability distributions of observations given the states, but only need to specify certain *feature functions*, which describe the correlations between the states and observations. CRFs may provide several benefits over the current models used in this thesis. Firstly, in the presence of probabilistic measurements, the probability distribution of observing a measurement given the state can not be modeled by any finite distribution. This thesis uses a marginalisation-based approach, which sums over all possible symbols weighted by their reported probabilities. However CRFs can address this in a more elegant way, where the probabilistic observations and the states can be correlated by selecting appropriate feature

functions. Secondly, in this thesis the measurements from multiple sensor systems are assumed to be conditionally independent given the states. However in practice it is possible that multiple sensor systems are corrupted by correlated errors, e.g. coexisting positioning systems may be influenced by a local distortion of the magnetic field. Using CRFs, we can lift this assumption by defining the feature functions that link the measurements from different systems together. Finally, in the case where a sensor system joins later than the others, it is more convenient to add it in the CRFs since the only thing we need to do is to define a new feature function, which correlates the new sensor measurements with the states, and the measurements from the others. Therefore in the future, we plan to use CRFs to extend our accuracy estimation framework, and compare them with the proposed approaches in this thesis.

Incorporate more types of prior knowledge: Another direction of future work is to incorporate more types of prior knowledge in accuracy estimation. In this thesis, we have only considered the priors that specify the state distribution at single timestamps, in future work, we would like to exploit priors that correlate multiple timestamps. For instance in the indoor positioning scenario, given the floor topology, a prior could be that "a user should not transit from one room to another without passing a corridor", which constrains the transitions between states at two consecutive timestamps (if the user is in a room at time t, then at $t + 1$ she has to either stay in the same room, or transit to the corridor), but does not explicitly specify the state distribution at either timestamp. Another example includes priors that model cardinality constraints, e.g. "the user typically spend more than 50% of the time in her office". One possible approach of incorporating more complex priors is to consider the constrained models, such as the work in [145] and [146]. In this case, the states and sensor observations will still be modeled with probabilistic graphical models. However the models are *constrained* by the priors, and during the inference or learning process the state sequences that fail to satisfy the constraints of the priors are eliminated. For instance, given the above transition and cardinality priors in the positioning example, inference will filter out the possible trajectories that have made impossible transitions and with less than 50% of the timestamps in the office. The probability mass is then redistributed to the other valid trajectories and thus the estimated states are biased accordingly. The accuracy of the sensor systems is evaluated with respect to the estimated states in the same way as discussed in this thesis. Another approach we would like to investigate is to assess the accuracy of sensor systems solely based on the priors. We plan to model the measurements from multiple sensor systems as probabilistic streams, and express the priors as queries. The accuracy of a sensor system is then defined as the likelihood of the priors been satisfied on the reported stream. For instance, the transition prior discussed above can be expressed as a boolean

query, asking the probability that a trajectory follows the valid transitions. This approach may work well in the cases where all sensor systems are reporting low quality measurements, because it does not estimate states based on reported measurements, but uses the priors as independent references to check the degrees of consistency of the systems. In practice, this accuracy metric can also be used to rank the sensor systems, e.g. one would always task the system that has the largest likelihood of satisfying the priors. We plan to incorporate the ranking techniques developed in our work [147], which are able to rank this likelihood efficiently.

Work with more complex sensor systems: Last but not least, in the future we would like to apply the proposed accuracy estimation approaches in more sensing scenarios, and extend them to work with more complex sensor systems. One scenario that we are interested in is indoor positioning. We have identified two larger sites for future experiments, one is the Ashmolean museum ($109m \times 89m$), and the other is the covered market ($108m \times 53m$). We plan to deploy different indoor positioning systems that rely on various sensing modalities, and use the proposed approaches to study their accuracy. The candidate positioning systems include but are not limited to: a) the WiFi-based positioning system proposed in [95], which is able to identify and mitigate the non-line-of-sight (NLOS) signal strength measurements; b) the encounter-based tracking system developed in [110], which uses radio encounters to pin down the trajectories produced by pedestrian dead reckoning; and c) the magneto-inductive tracking system in [5, 148], which uses magnetic fields to track the object positions in 3D space. Another scenario that we would like to study is related with the robotics research. For instance, the autonomous robots may be equipped with multiple types of sensors, such as cameras and laser rangers, and it is paramount to know their accuracy as robots move around the space. We plan to explore the possibility of extending the approaches in this thesis to that context, e.g. using data from laser sensors to infer the failure model of the vision sensors, and vice versa. It is also possible to incorporate external prior knowledge such as the environmental conditions or known landmarks, in similar ways as shown in this thesis. Another interesting research direction is to investigate how the sensor accuracy of a team of robots may impact their behaviours. For instance in a formation of multiple robots, it is desirable to task the robots with accurate sensors to explore the space, while using others as communication relays, to increase the overall performance and save energy.

Appendix A

Inference for Continuous State Space

Here we show the detailed derivation of the static and dynamic inference techniques for continuous state space. We consider the same setting formulated in Section 4.2.

A.1 Static Inference

We follow the formulation in Section 4.2, and consider the augmented model shown in Figure 4.12(b). We assume that the correlations between measurements and states are linear: $z_t^m = H_t^m x_t + v_t^m$. H_t^m is a matrix that maps the state space to the measurement space, and $v_t^m \sim \mathcal{N}(0, \Sigma_{z_t^m})$ carries the uncertainty in the observation z_t^m. Both H_t^m and v_t^m are considered as the known model parameters. Given the observed measurements $z_t^{1:M}$ and the prior ρ_t, the distribution of the estimated state $\hat{x}_t$ is:

$$p(\hat{x}_t) = p(x_t|z_t^{1:M}, \rho_t) \propto p(x_t|\rho_t) \prod_{m=1}^{M} p(z_t^m|x_t) \tag{A.1}$$

To incorporate the multiple sensor measurements observed at time t, we consider an iterative approach. Note that the product term in Equation (A.1) can be viewed as multiplying the distributions $p(z_t^m|x_t)$ to the distribution $p(x_t|\rho_t)$ one by one, and here we introduce a variable $\hat{x}_t^{(m)}$, indicating the estimated state after incorporating the m-th measurement, $1 \leq m \leq M$. Particularly, if there is a prior ρ_t at time t, we let $\hat{x}_t^{(0)} = p(\rho_t)$, to carry such prior knowledge. $\hat{x}_t^{(m)}$ is also Gaussian $\mathcal{N}(\mu_{\hat{x}_t^{(m)}}, \Sigma_{\hat{x}_t^{(m)}})$, which can be evaluated iteratively as:

$$\mu_{\hat{x}_t^{(m)}} = \mu_{\hat{x}_t^{(m-1)}} + K_t^{(m)}(\mu_{z_t^m} - H_t^m \mu_{\hat{x}_t^{(m-1)}}) \tag{A.2a}$$

$$\Sigma_{\hat{x}_t^{(m)}} = (I - K_t^{(m)} H_t^m)\Sigma_{\hat{x}_t^{(m-1)}} \tag{A.2b}$$

where I is the identity matrix, and $K_t^{(m)}$ is defined as:

$$K_t^{(m)} = [\Sigma_{\hat{x}_t^{(m-1)}}(H_t^m)^{\mathrm{T}}][H_t^m \Sigma_{\hat{x}_t^{(m-1)}}(H_t^m)^{\mathrm{T}} + \Sigma_{z_t^m}]^{-1} \tag{A.3}$$

From the above equations, we see that in each iteration, the mean of the estimated state $\mu_{\hat{x}_t^{(m)}}$ is a weighted linear combination of the previous estimated mean $\mu_{\hat{x}_t^{(m-1)}}$ and the mean of the next sensor observation $\mu_{z_t^m}$. The weight is determined by $K_t^{(m)}$, which depends on the covariances of the previous state estimate and the sensor observation. The covariance of the estimate state $\Sigma_{\hat{x}_t^{(m)}}$ also depends on both previous estimated covariance $\Sigma_{\hat{x}_t^{(m-1)}}$ and the covariance of the observed measurement $\Sigma_{z_t^m}$. Therefore conceptually this process refines the distribution of the estimated state by iteratively taking the information encoded in the M sensor measurements into account (weighted by their covariances), and the final estimated state $\hat{x}_t$ is given by the M-th variable $\hat{x}_t^{(M)}$.

The complexity of our algorithm is approximately $O(TMn_o^3)$, where n_o is dimension of the observation space, which comes from matrix inversion in Equation (A.3). The presence of probabilistic measurements does not increase the complexity, because the proposed inference approach redefines the model parameters v_t^m (the measurement noise) to carry the uncertainty encoded in the probabilistic measurements, without introducing additional elements to the model.

A.2 Dynamic Inference

Unlike static inference, the dynamic inference algorithm computes the estimated state $\hat{x}_t$ based on the M sequences of probabilistic measurements $z_{1:T}^{1:M}$ and the observed priors $\rho_{1:T}$. We assume the correlations between the states at different timestamps are linear: $x_t = F_t x_{t-1} + w_t$, where F_t is the transition matrix and w_t is the process noise, $w_t \sim \mathcal{N}(0, \Sigma_{w_t})$. Both F_t and w_t are assumed to be model parameters known a priori. F_t controls the transition from previous state x_{t-1} to the current state, and w_t determines the uncertainty associated with this state transition. As in the static case, we also assume that a measurement z_t^m observed at timestamp t linearly depends on the state x_t: $z_t^m = H_t^m x_t + v_t^m$. But unlike the existing linear dynamical systems which consider v_t^m as a model parameter, in our model v_t^m represents the uncertainty in the observed measurement z_t^m, $v_t^m \sim \mathcal{N}(0, \Sigma_{z_t^m})$.

At a given timestamp t, the algorithm takes the following three pieces of information into account to estimate $\hat{x}_t$:

- the estimated state $\hat{x}_{t-1}$ at the previous timestamp, which is evaluated based on all measurements and priors until $t-1$;
- the M probabilistic measurements $z_t^{1:M}$ and the prior ρ_t observed at time t; and
- the estimated state $\hat{x}_{t+1}$ at the future timestamp, which is evaluated based on all measurements and priors from $t+1$ to T.

Based on the above observation, this algorithm augments the standard Kalman smoother and processes the sensor measurements and priors in two passes. Since all the variables are Gaussian, the task of dynamic inference is also to compute the mean $\mu_{\hat{x}_t}$ and covariance $\Sigma_{\hat{x}_t}$ of the estimated state $\hat{x}_t$. For notation simplicity, we introduce a variable $\hat{x}_{t|1:t}^{(m)}$, which represents the estimated state given the measurements from all M sensor systems until time $t-1$, the priors $\rho_{1:t}$ until time t, and the measurements z_t^1, ..., z_t^m observed at time t. The distribution of the variable $\hat{x}_{t|1:t}^{(m)}$ is denoted as $\mathcal{N}(\mu_{t|1:t}^{(m)}, \Sigma_{t|1:t}^{(m)})$.

Particularly, we use the variable $\hat{x}_{t|1:t}^{(0)}$ to correlate the estimated states at time $t-1$ and t. As defined above, the variable $\hat{x}_{t-1|1:t-1}^{(M)}$ represents the estimated state which has already taken all M measurement at $t-1$ into account. In the following text we omit the superscript M, and use the form $\hat{x}_{t-1|1:t-1} \sim \mathcal{N}(\mu_{t-1|1:t-1}, \Sigma_{t-1|1:t-1})$. We define another variable $\hat{x}_{t|1:t-1}$, which represents the *predicted* state at t, given all the previous measurements and priors until time $t-1$, by applying the state transition on $\hat{x}_{t-1|1:t-1}$:

$$\mu_{t|1:t-1} = F_t \mu_{t-1|1:t-1} \tag{A.4a}$$

$$\Sigma_{t|1:t-1} = F_t \Sigma_{t-1|1:t-1} F_t^{\mathrm{T}} + \Sigma_{w_t} \tag{A.4b}$$

where F_t is the transition matrix, Σ_{w_t} is the covariance of the process noise. Then the mean and covariance of variable $\hat{x}_{t|1:t}^{(0)}$ can be computed by taking the prior ρ_t into account:

$$\mu_{t|1:t}^{(0)} = (\mu_{t|1:t-1}\Sigma_{\rho_t} + \mu_{\rho_t}\Sigma_{t|1:t-1})(\Sigma_{\rho_t} + \Sigma_{t|1:t-1})^{-1} \tag{A.5a}$$

$$\Sigma_{t|1:t}^{(0)} = (\Sigma_{\rho_t}\Sigma_{t|1:t-1})(\Sigma_{\rho_t} + \Sigma_{t|1:t-1})^{-1} \tag{A.5b}$$

This already gives the forward pass of the proposed dynamic inference algorithm: it scans the observed data from the beginning, and for each timestamp t, the algorithm first computes the starting point $\hat{x}_{t|1:t}^{(0)}$ as in Equations (A.5), and then iteratively computes the distribution of $\hat{x}_{t|1:t}^{(m)}$ $(1 \leq m \leq M)$ as:

$$\mu_{t|1:t}^{(m)} = \mu_{t|1:t}^{(m-1)} + K_t^{(m)}(\mu_{z_t^m} - H_t^m \mu_{t|1:t}^{(m-1)}) \tag{A.6a}$$

$$\Sigma_{t|1:t}^{(m)} = (I - K_t^{(m)} H_t^m)\Sigma_{t|1:t}^{(m-1)} \tag{A.6b}$$

where $K_t^{(m)}$ is given by:

$$K_t^{(m)} = [\Sigma_{t|1:t}^{(m-1)}(H_t^m)^{\mathrm{T}}][H_t^m \Sigma_{t|1:t}^{(m-1)}(H_t^m)^{\mathrm{T}} + \Sigma_{z_t^m}]^{-1} \tag{A.7}$$

In the second pass, the proposed algorithm processes the observed measurements and priors backwards from timestamp T to 1, and uses the previously computed $\mu_{t|1:t}$ and $\Sigma_{t|1:t}$ to compute the distribution of the estimated state $\hat{x}_t$, i.e. the posterior state distribution at time t given all measurements and priors observed during the time period $1:T$:

$$\mu_{\hat{x}_t} = \mu_{t|1:T} = G_t \mu_{t+1|1:T} + (I - G_t F_t)\mu_{t|1:t} \tag{A.8a}$$

$$\Sigma_{\hat{x}_t} = \Sigma_{t|1:T} = \frac{1}{2}(S_t + S_t^{\mathrm{T}}) \tag{A.8b}$$

Here G_t and S_t are given by:

$$G_t = (F_t \Sigma_{t|1:t})^{\mathrm{T}}(F_t \Sigma_{t|1:t} F_t^{\mathrm{T}} + \Sigma_{w_t})^{-1} \tag{A.9a}$$

$$S_t = G_t \Sigma_{t+1|1:T} F_t^{\mathrm{T}} + (I - G_t F_t)\Sigma_{t|1:t} \tag{A.9b}$$

where $\mu_{t+1|1:T}$ and $\Sigma_{t+1|1:T}$ are the mean and variance of the estimated state $\hat{x}_{t+1}$ at time $t+1$. Note that the starting point of the backward pass $\mu_{T|1:T}$ and $\Sigma_{T|1:T}$ are computed at the end of the forward pass.

The complexity of this dynamic inference algorithm is approximately $O(T(Mn_o^3 + n_x^3))$, where n_o is the dimension of the observation space, and n_x is the dimension of the state space. The term n_x^3 comes from the matrix inversion in Equation (A.9a). As in the static case, the incorporation of probabilistic observations does not increase the complexity because the uncertainty in measurements is carried by v_t^m, which are used as model parameters in the standard Kalman smoother.

Appendix B

Parameter Learning for Continuous State Space

The parameter learning problem of continuous latent variable models can be addressed by the general Expectation Maximisation (EM) scheme. Recall from Section 5.2 that the first step of the learning process is to derive the likelihood function $L(\theta; O, X)$ of the model parameter θ, which is the joint likelihood of the observed data O and the latent states X. The goal is to find the maximum likelihood estimate (MLE) $\hat{\theta}_{\text{ML}}$ of the model parameters, which maximise the likelihood of the observed data $\sum_X L(\theta; O, X)$. To achieve this, the EM scheme iteratively performs the *E*-step and *M*-step, where:

- The *E*-step derives the expected log likelihood function $Q(\theta', \theta)$ of the observed data. The expectation is taken with respect to the conditional distribution of the states X given the observed data O, under the current parameters θ:

$$Q(\theta', \theta) = E_{X|O,\theta}[\log L(\theta; O, X)] \tag{B.1}$$

- The *M*-step finds the parameters θ' that maximise the derived Q function of the observed data.

The EM scheme continues performing the above two steps until it reaches a local optimum. Now we show the detailed derivation for static and dynamic learning.

B.1 Static parameter learning

Observed data and model parameters: In static case, the observed data O_t at each timestamp is the M probabilistic sensor measurements $z_t^{1:M}$ from the coexisting sensor systems

and prior on state ρ_t, i.e. $O_t = \{z_t^{1:M}, \rho_t\}$. The model parameters considered in this case are the measurement models H_t^m, which map the states to the measurements: $z_t^m = H_t^m x_t + v_t^m$, where the variables v_t^m carry the uncertainty in the measurements.

Likelihood function: Given the observed data O_t and the state x_t, the likelihood function $L(\theta; O_t, x_t)$ is the joint probability distribution of the state and observed data:

$$\begin{aligned} L(\theta; O_t, x_t) &= p(z_t^{1:M}, \rho_t, x_t|\theta) \\ &= p(\rho_t, x_t|\theta) \prod_{m=1}^{M} p(z_t^m|\rho_t, x_t, \theta) \end{aligned} \tag{B.2}$$

The term $p(\rho_t, x_t)$ is the prior belief on state, and can be solely determined by the prior ρ_t (we assume that there is no other knowledge on $p(x_t)$). The term $p(z_t^m|\rho_t, x_t, \theta)$ is the probability distribution of observing the measurement z_t^m given the state x_t and prior ρ_t. Since both terms are Gaussian, the likelihood function $L(\theta; O_t, x_t)$ is also Gaussian.

***E*-step:** Given the derived likelihood function $L(\theta; O_t, x_t)$, in each learning iteration the *E*-step evaluates the expected log likelihood function $Q(\theta', \theta)$, with respect to the new parameters θ' that need to be learned:

$$\begin{aligned} Q(\theta', \theta) &= E_{x_t|O_t,\theta}[\log L(\theta'; O_t, x_t)] \\ &= E_{x_t|O_t,\theta}[\log p(\rho_t, x_t|\theta') + \sum_{m=1}^{M} \log p(z_t^m|\rho_t, x_t, \theta')] \end{aligned} \tag{B.3}$$

where the term $\log p(z_t^m|\rho_t, x_t, \theta')$ is:

$$\log p(z_t^m|\rho_t, x_t, \theta') = -\frac{1}{2}\log|\Sigma_{z_t^m}| - \frac{1}{2}(\mu_{z_t^m} - H_t^{m\prime} x_t)^{\mathrm{T}} \Sigma_{z_t^m}^{-1} (\mu_{z_t^m} - H_t^{m\prime} x_t) \tag{B.4}$$

since the correlation between the measurement z_t^m and state x_t is assumed to be linear and thus $p(z_t^m|\rho_t, x_t, \theta')$ is Gaussian. Note that here $\mu_{z_t^m}$ and $\Sigma_{z_t^m}$ are the mean and covariance of the observed measurement z_t^m, while $H_t^{m\prime}$ is the new parameters that need to be learned.

***M*-step:** The above derived $Q(\theta', \theta)$ is actually a function of the new parameters $H_t^{m\prime}$, and the goal of the *M*-step is to find the parameters $H_t^{m\prime}$ that maximise the Q function:

$$H_t^{m\prime} = \arg\max_{H_t^{m\prime}} Q(\theta', \theta) \tag{B.5}$$

Maximising Equation (B.3) with respect to $H_t^{m\prime}$, we have:

$$H_t^{m\prime} = \frac{\mu_{z_t^m} E[x_t^{\mathrm{T}}]}{E[x_t x_t^{\mathrm{T}}]} = \frac{\mu_{z_t^m} \mu_{\hat{x}_t}^{\mathrm{T}}}{\mu_{\hat{x}_t} \mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t}} \tag{B.6}$$

where the expectations $E[x_t^{\mathrm{T}}]$ and $E[x_t x_t^{\mathrm{T}}]$ are actually the mean and covariance of the estimated state ($\mu_{\hat{x}_t}^{\mathrm{T}}$ and $\Sigma_{\hat{x}_t}$) computed by static inference (as shown in Appendix A.1)

under the current model parameter H_t^m. Note that the priors ρ_t are taken into account during the evaluation of the $\mu_{\hat{x}_t}$ and $\Sigma_{\hat{x}_t}$, and thus can bias the learned model parameters.

The complexity of this static learning algorithm is approximately $O(ITM(n_x^3 + n_o^3))$, where I is the number of learning iterations, n_x is the dimension of state space, and n_o is the dimension of the observation space. This is because each learning iteration needs to infer the latent state x_t, which requires $O(TMn_o^3)$ as discussed in Appendix A.1, and the evaluation of $H_t^{m\prime}$ requires another matrix inversion which is approximately $O(n_x^3)$.

B.1.1 Dynamic parameter learning

Observed data and model parameters: In the dynamic case, the observed data O includes the M sequences of the probabilistic measurements $z_{1:T}^{1:M}$, and the sequence of priors $\rho_{1:T}$, i.e. $O = \{z_{1:T}^{1:M}, \rho_{1:T}\}$. Under the linear Gaussian assumption: $z_t^m = H_t^m x_t + v_t^m$ and $x_t = F_t x_{t-1} + w_t$, the model parameters θ that need to be learned include three elements: the transition model F_t, the covariance of the process noise vector Σ_{w_t}, and the measurement model H_t^m. F_t maps the current state vector to the next, Σ_{w_t} governs the inflated uncertainty in the state vector during state transitions, and H_t^m controls how the measurements of the m-th sensor system are correlated with the actual states. Since dynamic learning operates on the full sequences of observed data, here for simplicity we assume the model parameters F_t, Σ_{w_t} and H_t^m are independent of time t, and denote them as F, Σ_w, and H^m.

Likelihood function: Given the observed data O, the likelihood function $L(\theta; O, x_{1:T})$ is the joint probability of the observed data O and the state sequence $x_{1:T}$, given the model parameters θ:

$$\begin{aligned} L(\theta; O, x_{1:T}) &= p(z_{1:T}^{1:M}, \rho_{1:T}, x_{1:T}|\theta) \\ &= p(\rho_{1:T}, x_{1:T}|\theta) \prod_{m=1}^{M} p(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta) \end{aligned} \tag{B.7}$$

where the term $p(\rho_{1:T}, x_{1:T}|\theta)$ is the distribution of the prior and state sequence $x_{1:T}$, and the term $p(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta)$ is the distribution of the m-th sequence of observed measurements $z_{1:T}^m$ given the sequence of states and priors. The logarithm of those two terms can be computed as:

$$\log p(\rho_{1:T}, x_{1:T}|\theta) = \log p(\rho_1, x_1|\theta) + \sum_{t=2}^{T} \log p(\rho_t, x_t|\rho_{t-1}, x_{t-1}, \theta) \tag{B.8a}$$

$$\log p(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta) = \sum_{t=1}^{T} \log p(z_t^m|\rho_t, x_t, \theta) \tag{B.8b}$$

Note that since both the state transition $p(\rho_t, x_t|\rho_{t-1}, x_{t-1}, \theta)$ and the likelihood of measurement $p(z_t^m|\rho_t, x_t, \theta)$ are Gaussian, the likelihood function $L(\theta; O, x_{1:T})$ is also Gaussian.

***E*-step:** With the above derived likelihood function $L(\theta; O, x_{1:T})$, in each learning iteration the *E*-step evaluates the expected log likelihood function $Q(\theta', \theta)$, with respect to the new parameters $\theta' = \{F', \Sigma'_w, H^{m\prime}\}$ that need to be learned:

$$\begin{aligned} Q(\theta', \theta) =& E_{x_{1:T}|O,\theta}[\log L(\theta'; O, x_{1:T})] \\ =& E_{x_{1:T}|O,\theta}[\log p(\rho_{1:T}, x_{1:T}|\theta') + \sum_{m=1}^{M} \log p(z_{1:T}^m|\rho_{1:T}, x_{1:T}, \theta')] \\ =& E_{x_{1:T}|O,\theta}[\log p(\rho_{1:T}, x_{1:T}|F', \Sigma'_w) + \sum_{m=1}^{M} \log p(z_{1:T}^m|\rho_{1:T}, x_{1:T}, H^{m\prime})] \end{aligned} \tag{B.9}$$

which is broken into independent parts that depend on different parameters in θ'. The terms $\log p(\rho_{1:T}, x_{1:T}|F', \Sigma'_w)$ and $\log p(z_{1:T}^m, |\rho_{1:T}, x_{1:T}, H^{m\prime})$ can be evaluated as in Equation (B.8).

***M*-step:** The above derived $Q(\theta', \theta)$ is actually a function of the new parameters F', Σ'_w, and $H^{m\prime}$, and the goal of the *M*-step is to find the parameters that maximise the Q function. Maximising Equation (B.9) with respect to different parameters F', Σ'_w, and $H^{m\prime}$ independently, we have:

$$F' = \frac{\sum_{t=2}^{T} E[x_t x_{t-1}^{\mathrm{T}}]}{\sum_{t=1}^{T-1} E[x_t x_t^{\mathrm{T}}]} = \frac{\sum_{t=2}^{T} (\mu_{\hat{x}_t} \mu_{\hat{x}_{t-1}}^{\mathrm{T}} + \Sigma_{\hat{x}_{t-1}})}{\sum_{t=1}^{T-1} (\mu_{\hat{x}_t} \mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t})} \tag{B.10a}$$

$$\begin{aligned} \Sigma'_w &= \frac{1}{T-1} \sum_{t=1}^{T-1} \Big[E[x_{t+1} x_t^{\mathrm{T}}](I - F') + F'(E[x_t x_t^{\mathrm{T}}] F'^{\mathrm{T}} - E[x_t x_{t+1}^{\mathrm{T}}]) \Big] \\ &= \frac{1}{T-1} \sum_{t=2}^{T} [\mu_{\hat{x}_t} \mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t} - F'(\mu_{\hat{x}_t} \mu_{\hat{x}_{t-1}}^{\mathrm{T}} + \Sigma_{\hat{x}_t})^{\mathrm{T}}] \end{aligned} \tag{B.10b}$$

$$H^{m\prime} = \frac{\sum_{t=1}^{T} \mu_{z_t^m} E[x_t^{\mathrm{T}}]}{\sum_{t=1}^{T} E[x_t x_t^{\mathrm{T}}]} = \frac{\sum_{t=1}^{T} \mu_{z_t^m} \mu_{\hat{x}_t}}{\sum_{t=1}^{T} (\mu_{\hat{x}_t} \mu_{\hat{x}_t}^{\mathrm{T}} + \Sigma_{\hat{x}_t})} \tag{B.10c}$$

Here $\mu_{\hat{x}_t}$ and $\Sigma_{\hat{x}_t}$ are the mean and covariance of the estimated state $\hat{x}_t$, which is computed by our modified version of Kalman smoother as discussed in Appendix A.2, under the current model parameters F, Σ_w, and H^m. The priors ρ_t have been taken into consideration during the evaluation of $\mu_{\hat{x}_t}$ and $\Sigma_{\hat{x}_t}$, and thus can bias the learned model parameters. The complexity of this dynamic learning algorithm is $O(ITM(n_o^3 + n_x^3))$, where n_o, n_x are the dimensions of the observation and state space.

References

[1] I. Leontiadis, C. Efstratiou, C. Mascolo, and J. Crowcroft, "Senshare: Transforming sensor networks into multi-application sensing infrastructures," in *Proc. EWSN*, 2012.

[2] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. **77**, no. 2, 1989.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. **40**, no. 8, pp. 102 – 114, 2002.

[4] V. Dyo, S. A. Ellwood, D. W. Macdonald, A. Markham, C. Mascolo, B. Pásztor, S. Scellato, N. Trigoni, R. Wohlers, and K. Yousef, "Evolution and sustainability of a wildlife monitoring sensor network," in *Proc. SenSys*, 2010.

[5] A. Markham, N. Trigoni, S. Ellwood, and D. Macdonald, "Revealing the hidden lives of underground animals using magneto-inductive tracking," in *Proc . SenSys*, 2010.

[6] J. Beutel, S. Gruber, A. Hasler, R. Lim, A. Meier, C. Plessl, I. Talzi, L. Thiele, C. Tschudin, M. Woehrle, and M. Yuecel, "Permadaq: A scientific instrument for precision sensing and data recovery in environmental extremes," in *Proc. IPSN*, 2009.

[7] R. Huang, W.-Z. Song, M. Xu, N. Peterson, B. A. Shirazi, and R. LaHusen, "Real-world sensor network for long-term volcano monitoring: Design and findings," *IEEE Transactions on Parallel and Distributed Systems*, vol. **23**, no. 2, 2012.

[8] A. Goodney, D. Negandhi, K. Joshi, and Y. H. Cho, "Marina-scale water temperature sensing in real time using microtomography," in *Proc. ICDCSW*, 2011.

[9] T. W. Hnat, V. Srinivasan, J. Lu, T. I. Sookoor, R. Dawson, J. Stankovic, and K. Whitehouse, "The hitchhiker's guide to successful residential sensing deployments," in *Proc . SenSys*, 2011.

[10] K. Kapitanova, E. Hoque, J. A. Stankovic, K. Whitehouse, and S. H. Son, "Being smart about failures: assessing repairs in smart homes," in *Proc . UbiComp*, 2012.

[11] Wikipedia, "Electroencephalography — Wikipedia, the free encyclopedia," 2013. [Online; accessed 14-March-2013].

[12] Wikipedia, "Electrocardiography — Wikipedia, the free encyclopedia," 2013. [Online; accessed 14-March-2013].

[13] Wikipedia, "Electromyography — Wikipedia, the free encyclopedia," 2013. [Online; accessed 14-March-2013].

[14] R. F. Dickerson, E. I. Gorlin, and J. A. Stankovic, "Empath: a continuous remote emotional health monitoring system for depressive illness," in *Proc . Wireless Health*, 2011.

[15] S. Nirjon, R. F. Dickerson, Q. Li, P. Asare, J. A. Stankovic, D. Hong, B. Zhang, X. Jiang, G. Shen, and F. Zhao, "Musicalheart: A hearty way of listening to music," in *Proc . SenSys*, 2012.

[16] M. R. Ahsan, M. I. Ibrahimy, and O. O. Khalifa, "Emg signal classification for human computer interaction: a review," *European Journal of Scientific Research*, vol. **33**, no. 3, 2009.

[17] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *Communications Magazine, IEEE*, vol. **48**, no. 9, 2010.

[18] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application," in *Proc . SenSys*, 2008.

[19] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones," in *Proc . SenSys*, 2009.

[20] T. Choudhury, S. Consolvo, B. Harrison, J. Hightower, A. LaMarca, L. LeGrand, A. Rahimi, A. Rea, G. Bordello, B. Hemingway, *et al.*, "The mobile sensing platform: An embedded activity recognition system," *Pervasive Computing, IEEE*, vol. **7**, no. 2, 2008.

[21] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor wifi graphslam," in *Proc. ICRA*, 2011.

[22] B. Ferris, D. Fox, and N. Lawrence, "Wifi-slam using gaussian process latent variable models," in *Proc. IJCAI*, 2007.

[23] V. Moghtadaiee, A. Dempster, and S. Lim, "Indoor localization using fm radio signals: A fingerprinting approach," in *Proc. IPIN*, 2011.

[24] W. Grosky, A. Kansal, S. Nath, J. Liu, and F. Zhao, "Senseweb: An infrastructure for shared sensing," *MultiMedia, IEEE*, vol. **14**, no. 4, pp. 8–13, 2007.

[25] C. Efstratiou, I. Leontiadis, C. Mascolo, and J. Crowcroft, "Demo abstract: A shared sensor network infrastructure," in *Proc. SenSys*, 2010.

[26] H. Wen, Z. Xiao, N. Trigoni, and P. Blunsom, "On assessing the accuracy of positioning systems in indoor environments," in *Proc. EWSN*, 2012.

[27] S. Madden, "Intel lab data," 2004. [Online; accessed 14-March-2013].

[28] D. Kollar and N. Friedman, *Probabilistic graphical models: principles and techniques*. The MIT Press, 2009.

[29] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, 2nd ed., 2003.

[30] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 1. springer New York, 2006.

[31] R. Kindermann, J. L. Snell, *et al.*, *Markov random fields and their applications*, vol. 1. American Mathematical Society Providence, RI, 1980.

[32] A. V. Nefian, L. Liang, X. Pi, X. Liu, and K. Murphy, "Dynamic bayesian networks for audio-visual speech recognition," *EURASIP Journal on Advances in Signal Processing*, vol. 2002, no. 11, pp. 1274–1288, 1900.

[33] P. M. Jorge, A. J. Abrantes, and J. S. Marques, "On-line object tracking with bayesian networks," in *International Workshop on Image Analysis for Multimedia Interactive Systems, Lisbon*, 2004.

[34] D. Heckerman, "Bayesian networks for data mining," *Data mining and knowledge discovery*, vol. **1**, no. 1, pp. 79–119, 1997.

[35] D. Geiger and F. Girosi, "Parallel and deterministic algorithms from mrfs: Surface reconstruction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. **13**, no. 5, pp. 401–412, 1991.

[36] D. K. Panjwani and G. Healey, "Markov random field models for unsupervised segmentation of textured color images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. **17**, no. 10, pp. 939–954, 1995.

[37] D. Metzler and W. B. Croft, "A markov random field model for term dependencies," in *Proc . SIGIR*, 2005.

[38] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. **82**, no. D, pp. 35–45, 1960.

[39] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and computing*, vol. **10**, no. 3, pp. 197–208, 2000.

[40] N. L. Zhang and D. Poole, "A simple approach to bayesian network computations," 1994.

[41] J. Pearl, "Reverend bayes on inference engines: A distributed hierarchical approach," in *Proc . AAAI*, 1982.

[42] M. I. Jordan, ed., *Learning in graphical models*. Cambridge, MA, USA: MIT Press, 1999.

[43] A. Darwiche, "Recursive conditioning," *Artificial Intelligence*, vol. **126**, no. 1, pp. 5–41, 2001.

[44] D. Allen and A. Darwiche, "New advances in inference by recursive conditioning," in *Proc . UAI*, 2002.

[45] B. M. Bell, "The iterated kalman smoother as a gauss-newton method," *SIAM Journal on Optimization*, vol. **4**, no. 3, pp. 626–636, 1994.

[46] A. Doucet, N. d. Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. Springer, 1nd ed., 2001.

[47] H. Valpola and J. Karhunen, "An unsupervised ensemble learning method for nonlinear dynamic state-space models," *Neural computation*, vol. **14**, no. 11, pp. 2647–2692, 2002.

[48] Z. Ghahramani and G. E. Hinton, "Variational learning for switching state-space models," *Neural computation*, vol. **12**, no. 4, pp. 831–864, 2000.

[49] V. Bychkovskiy, S. Megerian, D. Estrin, and M. Potkonjak, "A collaborative approach to in-place sensor calibration," in *Proc. IPSN*, 2003.

[50] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich, "Collaborative signal and information processing: an information-directed approach," *Proceedings of the IEEE*, vol. **91**, no. 8, pp. 1199–1209, 2003.

[51] R. Nowak and U. Mitra, "Boundary estimation in sensor networks: Theory and methods," in *Proc. IPSN*, 2003.

[52] N. Cressie, "The origins of kriging," *Mathematical Geology*, vol. **22**, pp. 239–252, 1990.

[53] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*, vol. 1. MIT press Cambridge, MA, 2006.

[54] E. Ertin, "Gaussian process models for censored sensor readings," in *Statistical Signal Processing, 2007. SSP'07. IEEE/SP 14th Workshop on*, pp. 665–669, IEEE, 2007.

[55] M. A. Osborne, S. J. Roberts, A. Rogers, and N. R. Jennings, "Real-time information processing of environmental sensor network data using bayesian gaussian processes," *ACM Trans. Sen. Netw.*, vol. **9**, no. 1, 2012.

[56] B. Hummel, "Map matching for vehicle guidance," in *Dynamic and Mobile GIS: Investigating Space and Time*, CRC Press, 2006.

[57] J. Krumm, J. Letchner, and E. Horvitz, "Map matching with travel time constraints," in *SAE World Congress*, 2007.

[58] P. Newson and J. Krumm, "Hidden markov map matching through noise and sparseness," in *Proc. GIS*, 2009.

[59] J. Biagioni, T. Gerlich, T. Merrifield, and J. Eriksson, "Easytracker: automatic transit tracking, mapping, and arrival time prediction using smartphones," in *Proc . SenSys*, 2011.

[60] A. Thiagarajan, L. Ravindranath, H.Balakrishnan, S. Madden, and L. Girod, "Accurate, Low-Energy Trajectory Mapping for Mobile Devices," in *Proc . NSDI*, 2011.

[61] S. Guha, K. Plarre, D. Lissner, S. Mitra, B. Krishna, P. Dutta, and S. Kumar, "Autowitness: locating and tracking stolen property while tolerating gps and radio outages," in *Proc. SenSys*, 2010.

[62] J. Seitz, T. Vaupel, J. Jahn, S. Meyer, J. G. Boronat, and J. Thielecke, "A hidden markov model for urban navigation based on fingerprinting and pedestrian dead reckoning.," in *Proc. FUSION*, 2010.

[63] W. Chai, C. Chen, E. Edwan, J. Zhang, and O. Loffeld, "Ins/wi-fi based indoor navigation using adaptive kalman filtering and vehicle constraints," in *Proc. WPNC*, 2012.

[64] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *Proc. UbiComp*, 2008.

[65] P. Robertson, M. Angermann, B. Krach, and M. Khider, "Slam dance: Inertial-based joint mapping and positioning for pedestrian navigation," in *Proc. Inside GNSS*, 2010.

[66] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in *Proc. MobiCom*, 2012.

[67] B. Kanagal and A. Deshpande, "Online filtering, smoothing and probabilistic modeling of streaming data," in *Proc. ICDE*, 2008.

[68] T. Tran, C. Sutton, R. Cocci, Y. Nie, Y. Diao, and P. Shenoy, "Probabilistic inference over rfid streams in mobile environments," in *Proc. ICDE*, 2009.

[69] J. Letchner, C. Re, M. Balazinska, and M. Philipose, "Access methods for markovian streams," in *Proc. ICDE*, 2009.

[70] M. Coates, "Distributed particle filters for sensor networks," in *Proc. IPSN*, 2004.

[71] M. Paskin, C. Guestrin, and J. McFadden, "A robust architecture for distributed inference in sensor networks," in *Proc. IPSN*, 2005.

[72] S.-Z. Yu and H. Kobayashi, "A hidden semi-markov model with missing data and multiple observation sequences for mobility tracking," *Signal Processing*, vol. **83**, no. 2, pp. 235–250, 2003.

[73] K. Murphy, "A brief introduction to graphical models and bayesian networks."

[74] J. A. Snyman, *Practical mathematical optimization: an introduction to basic optimization theory and classical and new gradient-based algorithms*, vol. 97. Springer, 2005.

[75] Å. Björck, *Numerical methods for least squares problems*. Siam, 1996.

[76] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," 1952.

[77] L. R. Welch, "Hidden Markov Models and the Baum-Welch Algorithm," *IEEE Information Theory Society Newsletter*, vol. **53**, no. 4, 2003.

[78] X. Sun, L. Jin, and M. Xiong, "Extended kalman filter for estimation of parameters in nonlinear state-space models of biochemical networks," *PloS one*, vol. **3**, no. 11, p. e3758, 2008.

[79] G. Poyiadjis, A. Doucet, and S. S. Singh, "Maximum likelihood parameter estimation in general state-space models using particle methods," in *Proc of the American Stat. Assoc*, Citeseer, 2005.

[80] V. Peltola and A. Honkela, "Variational inference and learning for non-linear state-space models with state-dependent observation noise," in *Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on*, pp. 190–195, IEEE, 2010.

[81] R. D. Turner, M. P. Deisenroth, and C. E. Rasmussen, "State-space inference and learning with gaussian processes," in *International Conference on Artificial Intelligence and Statistics*, pp. 868–875, 2010.

[82] K. Whitehouse and D. Culler, "Calibration as parameter estimation in sensor networks," in *Proc . the 1st ACM international workshop on Wireless sensor networks and applications*, pp. 59–67, ACM, 2002.

[83] N. Kantas, S. Singh, and A. Doucet, "Distributed maximum likelihood for simultaneous self-localization and tracking in sensor networks," *Signal Processing, IEEE Transactions on*, vol. **60**, no. 10, pp. 5038–5047, 2012.

[84] D. Marinakis, G. Dudek, and D. J. Fleet, "Learning sensor network topology through monte carlo expectation maximization," in *Proc . ICRA*, 2005.

[85] M. Ding and X. Cheng, "Fault tolerant target tracking in sensor networks," in *Proc . MobiHoc*, 2009.

[86] D. Marinakis, N. MacMillan, R. Allen, and S. Whitesides, "Simultaneous localization and environmental mapping with a sensor network," in *Proc . ICRA*, 2011.

[87] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher, "On truth discovery in social sensing: a maximum likelihood estimation approach," in *Proc . IPSN*, 2012.

[88] M. S. Stankovic, S. S. Stankovic, and K. H. Johansson, "Distributed calibration for sensor networks under communication errors and measurement noise," in *Proc . CDC*, 2012.

[89] X. Chen, A. Edelstein, Y. Li, M. Coates, M. Rabbat, and A. Men, "Sequential monte carlo for simultaneous passive device-free tracking and sensor localization using received signal strength measurements," in *Proc . IPSN*, 2011.

[90] D. Wang, T. Abdelzaher, L. Kaplan, and C. Aggarwal, "Recursive fact-finding: A streaming approach to truth estimation in crowdsourcing applications," in *Proc . ICDCS*, 2013.

[91] M. Cypriani, F. Lassabe, P. Canalda, and F. Spies, "Wifi-based indoor positioning: Basic techniques, hybrid algorithms and open software platform," in *Proc. IPIN*, 2010.

[92] X. Li, "Rss-based location estimation with unknown pathloss model," *Wireless Communications, IEEE Transactions on*, vol. **5**, no. 12, pp. 3626–3633, 2006.

[93] S. Mazuelas, A. Bahillo, R. M. Lorenzo, P. Fernandez, F. A. Lago, E. Garcia, J. Blas, and E. J. Abril, "Robust indoor positioning provided by real-time rssi values in unmodified wlan networks," *Selected Topics in Signal Processing, IEEE Journal of*, vol. **3**, no. 5, pp. 821–831, 2009.

[94] Y. Wang, X. Jia, H. Lee, and G. Li, "An indoors wireless positioning system based on wireless local area network infrastructure," in *6th Int. Symp. on Satellite Navigation Technology Including Mobile Positioning & Location Services*, no. 54, 2003.

[95] Z. Xiao, H. Wen, A. Markham, N. Trigoni, P. Blunsom, and J. Frolik, "Identification and mitigation of non-line-of-sight conditions using received signal strength," in *Proc . WiMob*, 2013.

[96] P. Bahl and V. Padmanabhan, "Radar: an in-building rf-based user location and tracking system," in *Proc. INFOCOM*, 2000.

[97] A. LaMarca et al., "Place lab: device positioning using radio beacons in the wild," in *Proc. Pervasive*, 2005.

[98] M. Youssef and A. Agrawala, "The horus wlan location determination system," in *MobiSys*, 2005.

[99] M. B. Kjaergaard, "Indoor location fingerprinting with heterogeneous clients," *Pervasive Mob. Comput.*, vol. **7**, no. 1, pp. 31–43, 2011.

[100] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha, "Fm-based indoor localization," in *Proc. MobiSys*, 2012.

[101] M. Azizyan, I. Constandache, and R. Choudhury, "Surroundsense: Mobile phone localization via ambience fingerprinting," in *Proc. MobiCom*, 2009.

[102] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," in *Proc. MobiCom*, 2012.

[103] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *Computer Graphics and Applications, IEEE*, vol. **25**, no. 6, 2005.

[104] A. Jimenez, F. Seco, C. Prieto, and J. Guevara, "A comparison of pedestrian dead-reckoning algor. using a low-cost mems imu," in *WISP*, 2009.

[105] M. Susi, V. Renaudin, and G. Lachapelle, "Motion mode recognition and step detection algorithms for mobile phone users," *Sensors*, vol. **13**, no. 2, pp. 1539–1562, 2013.

[106] V. Renaudin, M. Susi, and G. Lachapelle, "Step length estimation using handheld inertial sensors," *Sensors*, vol. **12**, no. 7, pp. 8507–8525, 2012.

[107] B. Huyghe, J. Doutreloigne, and J. Vanfleteren, "3d orientation tracking based on unscented kalman filtering of accelerometer and magnetometer data," in *Proc. SAS*, 2009.

[108] Y. S. Suh, "Orientation estimation using a quaternion-based indirect kalman filter with adaptive estimation of external acceleration," *IEEE T. Instrum. and Measurement*, vol. **59**, no. 12, pp. 3296–3305, 2010.

[109] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury, "Did you see bob?: human localization using mobile phones," in *Proc. MobiCom*, 2010.

[110] A. Symington and N. Trigoni, "Encounter based sensor tracking," in *Proc. MobiHoc*, 2012.

[111] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A reliable and accurate indoor localization method using phone inertial sensors," in *Proc. UbiComp*, 2012.

[112] K. McCarthy, "Accuracy in positioning systems," in *Proc. The Motion Control Technology Conference*, 1991.

[113] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks," *Selected Areas in Communications, IEEE Journal on*, vol. **23**, no. 4, pp. 839–850, 2005.

[114] U. Tariq, K.-s. Lhee, and M.-P. Hong, "A quadtree based data accuracy scheme for wireless sensor networks," in *Proc. ICCIT*, 2007.

[115] E. Elnahrawy and B. Nath, "Cleaning and querying noisy sensors," in *Proc. WSNA*, 2003.

[116] D. Wang, L. Kaplan, T. Abdelzaher, and C. C. Aggarwal, "On scalability and robustness limitations of real and asymptotic confidence bounds in social sensing," in *Proc. SECON*, 2012.

[117] D. Hasenfratz, O. Saukh, and L. Thiele, "Model-driven accuracy bounds for noisy sensor readings," in *Proc. DCoSS*, 2013.

[118] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energy-accuracy aware localization for mobile devices," in *Proc. MobiSys*, 2010.

[119] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energy-accuracy trade-off for continuous mobile device location," in *Proc. MobiSys*, 2010.

[120] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, vol. **46**, no. 5, 1999.

[121] X. Yin, J. Han, and P. Yu, "Truth discovery with multiple conflicting information providers on the web," in *Proc . KDD*, 2007.

[122] J. Pasternack and D. Roth, "Knowing what to believe (when you already know something)," in *Proc . COLING*, 2010.

[123] D. Wang, T. Abdelzaher, H. Ahmadi, J. Pasternack, D. Roth, M. Gupta, J. Han, O. Fatemieh, H. Le, and C. Aggarwal, "On bayesian interpretation of fact-finding in information networks," in *Proc . FUSION*, 2011.

[124] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg, "Near optimal sensor placements: maximising information while minimising communication cost," in *Proc . IPSN*, 2006.

[125] A. Das and D. Kempe, "Sensor selection for minimising worst-case prediction error," in *Proc . IPSN*, 2008.

[126] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *J. Mach. Learn. Res.*, vol. **9**, 2008.

[127] G. M. P. O. Hare, C. Muldoon, and N. Trigoni, "Combining sensor selection with routing and scheduling in wireless sensor networks," in *Proc . DMSN in conjunction with VLDB*, 2011.

[128] D. Wang, H. Ahmadi, T. Abdelzaher, H. Chenji, R. Stoleru, and C. Aggarwal, "Optimizing quality-of-information in cost-sensitive sensor data fusion," in *Proc . DCOSS*, 2011.

[129] R. Van Meteren and M. Van Someren, "Using content-based filtering for recommendation," in *Proc. ECML Workshop*, 2000.

[130] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. WWW*, 2001.

[131] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Communications of the ACM*, vol. **40**, no. 3, pp. 66–72, 1997.

[132] L. Reznik and V. Kreinovich, "Fuzzy and probabilistic models of association information in sensor networks," in *Proc. FUZZ-IEEE*, 2004.

[133] E. Pacuit, "Voting methods," in *The Stanford Encyclopedia of Philosophy*, winter 2012 ed., 2012.

[134] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. **39**, no. 1, 1977.

[135] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *The Annals of Mathematical Statistics*, vol. **41**, no. 1, 1970.

[136] E. Süli and D. F. Mayers, *An introduction to numerical analysis*. Cambridge University Press, 2003.

[137] A. P. Pentland, "Interpolation using wavelet bases," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. **16**, no. 4, pp. 410–414, 1994.

[138] J. Gillard, "An historical overview of linear regression with errors in both variables," *Cardiff University School of Mathematics Technical Report*, 2006.

[139] T. Söderström, "Errors-in-variables methods in system identification," *Automatica*, vol. **43**, no. 6, pp. 939–958, 2007.

[140] "x-io technologies," 2013. [Online; accessed 14-Oct-2013].

[141] S. Madgwick, A. J. L. Harrison, and R. Vaidyanathan, "Estimation of imu and marg orientation using a gradient descent algorithm," in *Proc . ICORR*, 2011.

[142] L. Bruno and P. Robertson, "Wislam: Improving footslam with wifi," in *Proc. IPIN*, 2011.

[143] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tinydb: An acquisitional query processing system for sensor networks," *TODS*, vol. **30**, no. 1, pp. 122–173, 2005.

[144] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, 2001.

[145] S. T. Roweis, "Constrained hidden markov models.," in *Proc. NIPS*, 1999.

[146] H. Christiansen, C. t. Have, O. t. Lassen, and M. Petit, "Inference with constrained hidden markov models in prism," *Theory Pract. Log. Program.*, vol. **10**, no. 4-6, 2010.

[147] D. Olteanu and H. Wen, "Ranking in probabilistic databases: Complexity and efficient algorithms," in *Proc. ICDE*, 2012.

[148] A. Markham and N. Trigoni, "Magneto-inductive networked rescue system (miners): taking sensor networks underground," in *Proc. IPSN*, 2012.