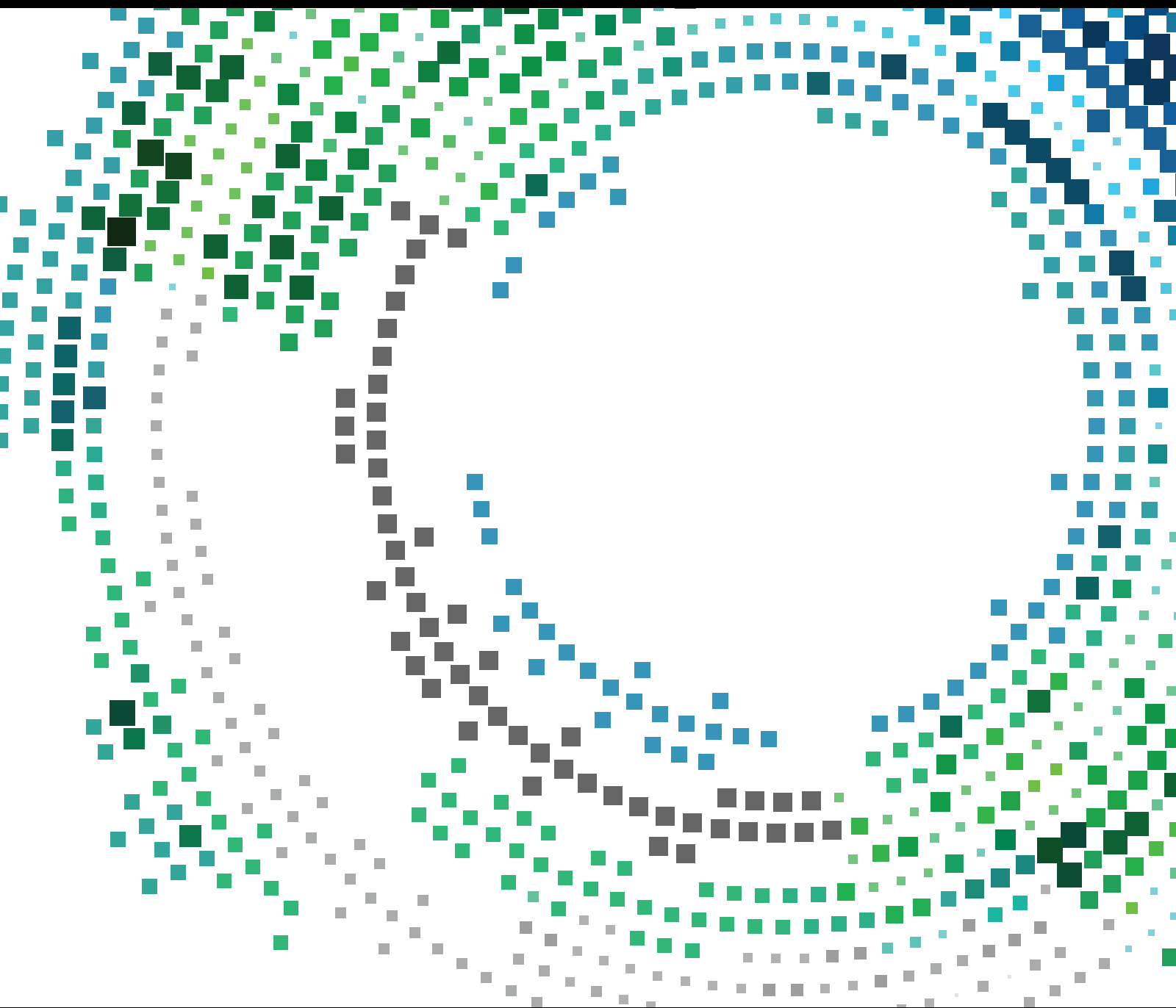
Mobile Information Systems

# Mobile Geospatial Computing Systems for Ubiquitous Positioning

Guest Editors: Liang Chen, Olivier Julien, Elena-Simona Lohan, Gonzalo Seco-Granados, and Ruizhi Chen



# Mobile Geospatial Computing Systems for Ubiquitous Positioning

Mobile Information Systems

# Mobile Geospatial Computing Systems for Ubiquitous Positioning

Lead Guest Editor: Liang Chen
Guest Editors: Olivier Julien, Elena-Simona Lohan, Gonzalo Seco-Granados, and Ruizhi Chen

# Editorial Board

# Contents

**Mobile Geospatial Computing Systems for Ubiquitous Positioning**
Liang Chen, Olivier Julien, Elena-Simona Lohan, Gonzalo Seco-Granados, and Ruizhi Chen
Editorial (2 pages), Article ID 9138095, Volume 2018 (2018)

**A Method to Incorporate Floor Plan Constraints into Indoor Location Tracking: A Voronoi Approach**
John D. Hobby and Marzieh Dashti
Research Article (11 pages), Article ID 5303616, Volume 2018 (2018)

**Positioning Using Terrestrial Multipath Signals and Inertial Sensors**
Christian Gentner, Robert Pöhlmann, Markus Ulmschneider, Thomas Jost, and Siwei Zhang
Research Article (18 pages), Article ID 9170746, Volume 2017 (2018)

**Semantic Labeling of User Location Context Based on Phone Usage Features**
Helena Leppäkoski, Alejandro Rivero-Rodriguez, Sakari Rautalin, David Muñoz Martínez, Jani Käppi, Simo Ali-Löytty, and Robert Piché
Research Article (21 pages), Article ID 3876906, Volume 2017 (2018)

**An Efficient Normalized Rank Based SVM for Room Level Indoor WiFi Localization with Diverse Devices**
Yasmine Rezgui, Ling Pei, Xin Chen, Fei Wen, and Chen Han
Research Article (19 pages), Article ID 6268797, Volume 2017 (2018)

**An Online Solution of LiDAR Scan Matching Aided Inertial Navigation System for Indoor Mobile Mapping**
Xiaoji Niu, Tong Yu, Jian Tang, and Le Chang
Research Article (11 pages), Article ID 4802159, Volume 2017 (2018)

**Ubiquitous and Seamless Localization: Fusing GNSS Pseudoranges and WLAN Signal Strengths**
Philipp Richter and Manuel Toledano-Ayala
Research Article (16 pages), Article ID 8260746, Volume 2017 (2018)

**Assessment of Smartphone Positioning Data Quality in the Scope of Citizen Science Contributions**
Angel J. Lopez, Ivana Semanjski, Sidharta Gautama, and Daniel Ochoa
Research Article (11 pages), Article ID 4043237, Volume 2017 (2018)

**Feasibility Study of Using Mobile Laser Scanning Point Cloud Data for GNSS Line of Sight Analysis**
Yuwei Chen, Lingli Zhu, Jian Tang, Ling Pei, Antero Kukko, Yiwu Wang, Juha Hyyppä, and Hannu Hyyppä
Research Article (11 pages), Article ID 5407605, Volume 2017 (2018)

**An Analysis of Impact Factors for Positioning Performance in WLAN Fingerprinting Systems Using Ishikawa Diagrams and a Simulation Platform**
Keqiang Liu, Yunjia Wang, Lixin Lin, and Guoliang Chen
Research Article (20 pages), Article ID 8294248, Volume 2017 (2018)

Hindawi
Mobile Information Systems
Volume 2018, Article ID 9138095, 2 pages
https://doi.org/10.1155/2018/9138095

*Editorial*

# Mobile Geospatial Computing Systems for Ubiquitous Positioning

**Liang Chen**,[1] **Olivier Julien**,[2] **Elena-Simona Lohan**,[3] **Gonzalo Seco-Granados**,[4] **and Ruizhi Chen**[1]

[1]*State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China*
[2]*Signal Processing and Navigation (SIGNAV) Research Group, Telecom Laboratory, Ecole Nationale de l'Aviation Civile (ENAC), 31055 Toulouse, France*
[3]*Department of Electronics and Communications Engineering, Tampere University of Technology (TUT), P.O. Box 553, 33101 Tampere, Finland*
[4]*Department of Telecommunications and Systems Engineering, Universitat Autònoma de Barcelona (UAB), 08193 Bellaterra, Spain*

Correspondence should be addressed to Liang Chen; l.chen@whu.edu.cn

Received 20 May 2018; Accepted 20 May 2018; Published 20 June 2018



Geospatial computing involves using computing devices and sensors to acquire, process, analyze, manage, and visualize geospatial data. However, the tasks of geospatial computing systems are computationally demanding in terms of computation power, data storage capacity, and memory space. With the recent developments in mobile computing and sensor technologies, mobile devices are able to meet the demanding requirements for geospatial computing. As a consequence, mobile geospatial computing systems (MGCSs) emerge and are developed rapidly. Currently, MSCSs have shown their significant importance in facilitating our daily life in many areas, for example, personal navigation based on virtual reality, mobile games based on mixed reality, self-driving car, unmanned taxi-service, and many location-based services. It should be noted that among all the functionality within the MSCSs, the ubiquitous positioning is one of the key supporting technologies. The accuracy of the computed positioning largely affects the quality of service in other applications related to MSCSs, for example, the mobile mapping and mobile geographical information systems (GIS).

In the past decades, a variety of wireless positioning technologies have been developed, which include GNSS (Global Navigation Satellite Systems), methods to exploit signal-of-opportunities, such as WiFi, RFID, cellular LTE/4G, UWB, WLAN, Bluetooth, digital TV, acoustic/millimeter-wave/light signals, and the hybrid solutions encompassing inertial measurement unit, sonar, laser, infrared (IR), magnetic field, camera, and so on. However, there are still many challenges in emerging applications, which need to be solved, for example, navigation in indoor environments, the security of the navigation systems to defend against threats, the data fusion of all source positioning and navigation, and so on. To resolve such challenges, the sensor-rich and computation enabling MSCSs may offer new potential.

The special issue aims to publish the most recent advances in the usage of the MSCSs to improve the quality of the ubiquitous positioning, as well as the development of innovative methods to provide more accurate and reliable positioning for MSCSs. We received a total of 21 submissions, spanning a range of topics from user location, context detection, multisensor-based indoor localization and mapping, human travel behavior studies, point cloud data-assisted context sensing, to indoor map assisted localization. After a thorough peer review process, 9 articles are selected, which are summarized below.

The first article in this special issue entitled "Positioning Using Terrestrial Multipath Signals and Inertial Sensors" authored by C. Gentner et al. exploit multipath propagation for position estimation of mobile receivers. A particle filtering-based Channel-SLAM is proposed, which fuses heading information of an inertial measurement unit (IMU) to improve the position accuracy.

A. J. Lopez et al. in the second article "Assessment of Smartphone Positioning Data Quality in the Scope of Citizen Science Contributions" analyze the completeness aspects of the data quality using GNSS data collected through smartphones from the campaigns. The results can be used in human travel behavior studies.

In the third article, "An Efficient Normalized Rank Based SVM for Room Level Indoor WiFi Localization with Diverse Device," L. Pei et al., study the problem of received signal strength index (RSSI) variation of different devices in WiFi fingerprinting-based indoor localization. A normalized rank-based support vector machine classifier (NR-SVM) is presented, and the validation of the algorithm has been tested by using 16 different devices in a shopping mall.

X. Niu et al. in the fourth article, "An Online Solution of LiDAR Scan Matching Aided Inertial Navigation System for Indoor Mobile Mapping" present online navigation algorithm for indoor mobile mapping with LiDAR and IMU integrated unmanned ground vehicle (UGV) system.

The fifth article, entitled "Semantic Labeling of User Location Context Based on Phone Usage Features" authored by H. Leppäkoski et al. proposes a machine learning-based method to detect the user's home, work, and other visited places by utilizing mobile phone usage features.

The sixth article considers incorporating map constraints into localization algorithms with the aim to reduce the uncertainty of walking trajectories and enhance location accuracy. J. Hobby and M. Dashti propose a method to generate indoor maps from CAD floor plans and an adapted map-filtering algorithm for indoor navigation. More details can be found in "A Method to Incorporate Floor Plan Constraints into Indoor Location Tracking: A Voronoi Approach."

The seventh article entitled "Ubiquitous and Seamless Localization: Fusing GNSS Pseudoranges and WLAN Signal Strengths" authored by P. Richter and M. Toledano-Ayala presents seamless positioning based on a particle filter tightly integrated GNSS pseudoranges and WLAN received signal strength indicators (RSSIs).

K. Liu et al. in the eighth article, entitled "An Analysis of Impact Factors for Positioning Performance in WLAN Fingerprinting Systems using Ishikawa Diagrams and a Simulation Platform" consider the impact factors on the positioning accuracy during the procedure of the indoor RSSI fingerprint localization. The factors include, for example, the access points (AP) density, signal propagating attenuation factor, and the reference points (RPs) density etc.

Finally, Y. Chen et al. in the ninth paper, entitled "Feasibility Study of Using Mobile Laser Scanning Point Cloud Data for GNSS Line of Sight Analysis" detect the line of sight condition by using mobile laser scanning point cloud in the urban canyon scenarios to enhance the position accuracy.

Our guest editorial team would like to thank all the authors and reviewers for their contribution to this special issue.

*Liang Chen*
*Olivier Julien*
*Elena-Simona Lohan*
*Gonzalo Seco-Granados*
*Ruizhi Chen*

Hindawi
Mobile Information Systems
Volume 2018, Article ID 5303616, 11 pages
https://doi.org/10.1155/2018/5303616

*Research Article*

# A Method to Incorporate Floor Plan Constraints into Indoor Location Tracking: A Voronoi Approach

**John D. Hobby**[1] **and Marzieh Dashti**[2]

[1]*Nokia Bell Labs, Murray Hill, NJ, USA*
[2]*Accenture The Dock, Ireland*

Correspondence should be addressed to Marzieh Dashti; marzieh.dashti@accenture.com

Received 12 January 2017; Revised 9 May 2017; Accepted 17 December 2017; Published 20 February 2018

Academic Editor: Olivier Julien



Indoor localization has attracted a lot of research effort in recent years due to the explosion of indoor location-based service (LBS) applications. Incorporating map constraints into localization algorithms reduces the uncertainty of walking trajectories and enhances location accuracy. Suitable maps for computer-aided localization algorithms are not readily available, and hence most researchers working on localization solutions manually create maps for their specific localization scenarios. This paper presents a method of generating indoor maps suitable for localization algorithms from CAD floor plans. Our solution is scalable for mass-market LBS deployment. We also propose an adapted map-filtering algorithm that utilizes map information extracted from CAD floor plans. We evaluate the performance of our solution via real-world Wi-Fi RF measurements.

## 1. Introduction

Accurate and robust indoor localization is a key enabler for numerous emergency and commercial services. Some examples of indoor location-based services (LBS) are as follows [1]: locate people on a map and navigate them to their destinations in shopping malls, airports, hospitals, and museums; recommend nearest business or services such as ATMs, retail stores, restaurants, and social events; produce location-based advertising; find and track stolen or lost objects; and monitor human activities. When displaying localization and tracking results, it is natural to superimpose the coordinates on the building floor plan, as an absolute location value is not of much use without its relation to the surrounding area map. In indoor LBS applications, the location data are visualized on a user-friendly building map displayed on the user device's screen. Besides the requirement of maps for location data visualization, map information can be utilized to enhance location accuracy and reduce uncertainty of walking trajectories.

Many indoor localization and tracking algorithms rely to a certain extent on map-based filtering methods to bound drift and noise-induced errors. These algorithms are most commonly based on particle filters [2]. The basic idea is fairly straightforward: the user's trajectory is described by a set of particles. The particle distribution models the measured trajectory as well as errors of the measurement systems [3]. Particles are not allowed to move to positions that violate the map constraints. For example, particles are not allowed to cross directly through walls. Particles that transit through such obstacles are down weighted or resampled [4].

Different ways of dealing with floor plan information are presented in the literature. In [5], particles crossing walls are eliminated by using a bitmap-based map matching algorithm. The state of the system and consequently the state of each particle consists of two pixel coordinates $x$ and $y$ on a bitmap. When the path of a particle (implemented as line painting on the bitmap) includes a nonwhite pixel, a collision with an obstacle has occurred and the weight of the particle is set to zero. Note that this requires a white representation of walkable space on the bitmap which includes, for example, the removal of doors or room names possibly depicted on the bitmap. In [6], the particle filter builds on a mixed graph/free space representation of indoor environments. While hallways, stair cases, and elevators are represented by edges in a graph, areas such as rooms are

represented by bounded polygons. Using this representation, both constrained motion such as moving down a hallway or going upstairs and less-constrained motion through rooms and open spaces are modeled. In [7], the authors assume that a floor plan is a set of walls, each described by the coordinates of the end points of the wall. Doors are modeled as gaps in the walls.

Generally speaking, the maps are basically drawings for human consumption, and they present some difficulties when used for algorithmic analysis. And, therefore, most researchers working on localization solutions manually create the required maps for their specific testing scenarios. While this approach is valid to test the performance of map-filtering algorithms, it cannot be scaled for high-volume commercial LBS applications.

As stated above and also argued in [3, 8], the requirements for the maps used for visualization and location estimation differ [8]. The lack of maps that are suitable for both, visualization and map-based filtering, is one of the main challenges for the mass-market deployment of indoor positioning systems. The building floor plan maps are commonly designed using computer-aided design (CAD) commercial software applications. Hence we aim at converting the floor plan CAD file in a format which is suitable as an input to the map-filtering algorithms. To the best of our knowledge, the only work that talks about this important problem is [3]. The authors present a parser that analyses standard CAD files to extract topological map information. This information is used to create an object-based map optimized for localization and tracking applications.

This paper presents a scalable method of generating indoor maps suitable for localization algorithms from CAD data. We discuss how to handle interior walls when using indoor tracking to locate smart phones or other devices based on technologies such as Wi-Fi signal strength. While we address the issues discussed in [3] with a new method, our focus is more on the need to get room polygons, which leads to planar subdivisions and the need to locate "doorway features." These are relatively narrow openings that should be treated like doorways. Another issue not discussed in [3] is that the building description contains more detail than is needed for tracking. We tried simplification strategies based on replacing polygonal lines that are well approximated by a single straight line segment. This speeds up the tracking algorithm with no significant effect on accuracy. We analyze the performance of our proposed algorithms using real-world RF fingerprint measurement data.

The remainder of this paper is structured as follows: in Section 2, we describe how to extract walls and room polygons' information from a floor plan CAD file. Section 3 describes our proposed tracking algorithm which incorporates floor plan constraints. Section 4 illustrates the tracking results using real-world Wi-Fi RF measurements taken in an enterprise building in Dublin. Section 5 concludes the paper.

## 2. Finding Walls and Room Polygons

There are at least two reasons for the location-finding algorithm itself to consider the floor plan and interior walls:

(1) Since people cannot walk through walls, it seems desirable for the tracking algorithm to forbid such trajectories.

(2) Depending on materials used, carrier frequency, and angle of incidence, walls tend to reflect a portion of the signal. Hence the tracking algorithm should expect walls to cause discontinuous changes in signal strength.

For Reason 2, it may suffice to have a set of line segments that the device being tracked is forbidden to cross. We extracted the line segments coordinates from our test bed building CAD floor plan and created an *edge list file*. However, Reason 2 suggests a dividing the building interior into a set of disjoint polygons, where some of the edges are designated as "doors" (in general, any part of a room boundary that the tracked device may pass through). We recorded the room polygons information into a *room polygon file*. In this section, we present in detail how to handle walls and room polygons.

Suppose the building floor plan is available in some machine-readable form (e.g., AutoCad®), and it is composed of *objects* such as "line," "polyline," "circular arc," and "text." Furthermore, it is divided into *layers*, and some layers may have multiple instances of "blocks." For example, the "furniture" layer may have "chair" blocks, where all chair blocks are affine transformations of each other.

The floor plans are basically drawings for human consumption, and they present some difficulties when used for algorithmic analysis. We have data for one floor of a $139 \times 101$ meter building in Dublin, Ireland. Our sample data had the following problems:

(1) Walls needed for room-based analysis are not on clearly defined layers; for example, some room dividers are in the furniture layer.

  (i) It helps to look at object names and block names as well as layer names.

  (ii) Objects named "Hatch" and blocks named "Door" are not walls.

(2) Heuristic tests are necessary to recognize features such as swinging doors and partitions in the furniture layer.

  (i) Instead of looking at rectangle aspect ratio, it is safer to compare the perimeter $P$ to the square root of the enclosed area $A$. For instance, $P/\sqrt{A} > 20/3$ implies a rectangle aspect ratio $> 9$, and it is meaningful for nonrectangles.

(3) Features such as door jambs and window frames provide numerous instances where different objects should share common $x$ or $y$ coordinates. These often do not match exactly.

  (i) Errors of this type are typically between 0.3 mm and 1 mm.

The object is to construct a set of polygons that accurately reflect the dimensions of the building (and its interior

Table 1: Parameters that control the door recognizer.

| Parameter | Value | Explanation |
|---|---|---|
| AuxDoorLim | 1 | 4th power of maximum-allowed aux door jamb error versus door length |
| MaybeAspect | 8 | Polyline furniture beyond this aspect ratio is a partition |
| DoorHrange | 80 | Swinging parts of a door must have handles in this interval |
| DoorH1range | 5 | Door rectangle sides must have handles in this interval |
| DJamAspect | 0.75 | Width versus length for excluded door jamb rectangle |
| DoorMinArc | Pi/4 | Minimum angle subtended by a door arc |
| DoorMaxArc | 7*Pi/12 | Maximum angle subtended by a door arc |
| DoorAspect | 6 | Minimum aspect ratio for a door rectangle |
| DoorWdErrFrac | 0.2 | Maximum door rectangle width error versus widest width |
| DoorWdErrvsL | 0.03 | Maximum door rectangle width error as fraction of length |
| DoorOffvsWd | 0.2 | Maximum door end squareness error distance versus widest width |
| DoorOffvsL | 0.03 | Maximum door end squareness error distance versus length |
| Bool Door1lineOK | True | May a door have one line rather than 2 parallel lines? |
| DoorShortArcvsL | 0.15 | Tolerance for door arc short of latch versus door length |
| DoorPastArcvsL | 0.20 | Tolerance for door arc past latch versus door length |
| DoorShortArcMin | 0.01 | Minimum of both tolerance for door arc short of latch versus door length |
| DoorPastArcMin | 0.05 | Minimum of both tolerance for door arc past latch versus door length |
| DoorMisfitArc | 0.03 | Fraction of door length by which door latch may misfit arc |
| DoorArcOffCtr | 0.05 | Fraction of door length for hinge versus arc center-squared error |
| DoorMinOffCtr | 0.026 | Minimum door length fractions for hinge versus squared arc center error |

walls) and show how it is possible to move from room to room. More precisely, we need a *planar subdivision*. A planar subdivision has vertices, edges, and faces, and the edges that meet at a common vertex are sorted by angle modulo 360 degrees. Furthermore, two faces meet at each edge, and it is possible to walk through the data structure in order to find the sequence of edges that describe any given face as a polygon [9]. For our application, the edges are polygonal lines, and certain segments are marked as doors (meaning that the tracking algorithm assumes its target can pass through).

Section 2.1 briefly outlines the heuristics for deciding which parts of the building description are walls and doorways. Then Section 2.2 explains how to convert the wall segments into a planar subdivision that defines rooms. Finally, Section 2.3 explains how to recognize places where doorway segments should be inserted so as to prevent large complicated regions from being erroneously treated as one room. This can be difficult to automate, so we also allow door segments to be added manually.

The generation of indoor maps from CAD data in the face of Problems 1-2 has been studied before in [3]. However, we focus more strongly on the need to get room polygons, and this leads to planar subdivisions (Section 2.2) and the door segment insertion problem (Section 2.3).

*2.1. Heuristics for Identifying Walls.* Heuristics are needed to recognize walls and doorways in the building description. To the extent that these could vary from one building to the next, the implementation needs to be as table-driven as possible; that is, customizing it to a new building should require new parameter values, not new algorithms. Although our sample data have "Door" blocks, only a small fraction of the doors are identified this way. Hence we have a Boolean expression for whether an object is worth keeping, and this is followed by logic that identifies doors. The door logic is controlled by numeric parameters as shown in Table 1.

In the tabular form of the Boolean decision, one table gives (layer name and decision index) pairs, and there are other such tables for object names and block names. The Boolean expression for whether to keep an object or treat it as junk is determined by another table where each node has a query value $q$ for one of the decision indices as well as "instructions" for $\leq q$ and $> q$. Each such "instruction" is either the index for another node in the table or a final decision ("Keep it," "Junk," or "Keep it if beyond MaybeAspect"). The main table has 6 of these nodes, and the table that gives decision indices for layer names considers 6 layer names and produces decision indices 0, 1, or 2.

Figure 1 exemplifies the input to the door-recognizing heuristics. The primary indication for the presence of a door is an elongated rectangle with a circular arc nearby. It is necessary to eliminate the picture of the swinging door and insert a segment marked "Door" that corresponds to the door in the closed position. Figure 1(a) has two horizontal lines that are near the desired closed-door segment. There are parameters in Table 1 that determine how to recognize such door jamb segments so that they can be removed.

*2.2. Building a Planar Subdivision.* The reason for wanting a planar subdivision is to have a complete set of room polygons with clearly defined geometry, where we can easily identify the two rooms connected by each doorway. Since

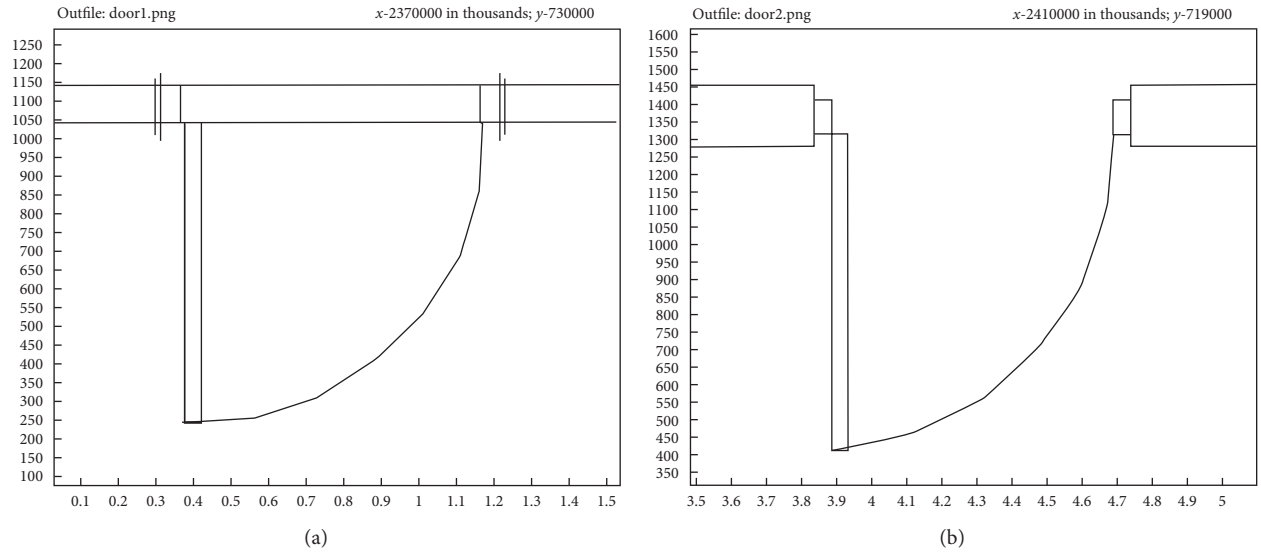



Figure 1: Typical door drawings that must be recognized by the heuristics. Approximating circular arcs by polygonal lines is artifact from the software used to create this figure. The horizontal scale is in meters, and the vertical scale is similar (but indicated in millimeters).

the data structure from [9] is well known, we need to only explain how to obtain a good set of wall and door segments.

We start by approximating ellipses, circular arcs, and splines by polygonal lines. Then we find all segment intersections and insert explicit intersection points using snap rounding as explained in [10]. There is also a need to be no overlapping segments, but having all intersections explicit makes it trivial to find them and remove duplicates.

It should be noted that buildings often allow one to walk from one room to another without encountering a swinging door. It is necessary to add door segments to the data structure in order to prevent large complicated regions from being erroneously treated as single rooms. Section 2.3 will explain how to do this. We perform this step immediately after replacing swinging doors by door segments.

We now have everything that is needed in order to construct a planar subdivision, but Figure 1 suggests a problem. Since walls are drawn with two parallel lines (and sometimes additional internal details), the planar subdivision will have a large number of faces that correspond interiors of walls. We simply need to postprocess the planar subdivision so that the remaining faces are actual rooms.

In order to distinguish room polygons from polygons that are just within walls, we use $2A/P$ as a generalized notion of average width, where $A$ is the polygon area and $P$ is the perimeter. We simply reject rooms for which $2A/P < D_0$, where $D_0$ is the minimum door size. To compute the minimum door size, we use the heuristics from Section 2.1 to find as many doors as possible, then compute the length of each door segment and find the 5th percentile.

*2.3. Voronoi Diagrams for Doorway Features.* The door-finding heuristics from Section 2.1 eliminate most of the swinging door pictures and replace them with doorway segments. Figure 2(a) shows a portion of the result. The large room in the middle has a doorway but no swinging door, so no door segment was inserted. Jagged line $a$ shows where the segment should be, and jagged lines $b, c,$ and $d$ show where more door segments are desirable.

The places with missing doorways are quite similar to the "stroke-like features" from [11]. As in that paper, the process of finding such features is based on the Voronoi diagram for line segments. The intuitive idea is that such a diagram finds centers of hallways but generalizes the notion of hallway width so that missing doorways appear as places of constricted width.

Given a collection of line segments, we can divide the surrounding space into regions based on which the line segment is closest. The borders of such regions are composed of straight lines and portions of parabolas. Any point $P$ on the boundary between two of these Voronoi regions is equidistant from two input segments. Suppose $Q_L$ and $Q_R$ are the points where those segments most closely approach $P$. An important idea from [11] is that the *opposition angle* $Q_LPQ_R$ can be used to prune the Voronoi diagram. The portions of the Voronoi diagram where the opposition angle is close to 180° are the "generalized hallways," and the "generalized hallway width" is the Euclidean distance from $P$ to $Q_L$ or $Q_R$ (it does not matter which).

Throwing away portions of the Voronoi diagram where the opposition angle is <146° produces curvilinear paths that track the centers of hallways as well as the centers of walls and also the missing doorway features that we are looking for. This is how we generated the lines that were added to Figure 2(a) in order to generate Figure 2(b). As long as the threshold is >120°, it is impossible for three or more of the pruned Voronoi segments to meet. Hence the pruned Voronoi is composed of curvilinear paths for which the opposition angle is a function of distance along the path.

Figure 3 suggests how to use the pruned Voronoi paths to recognize openings where door segments are needed. When a Voronoi path goes through a doorway-type opening, the opposition angle increases and then decreases. We can

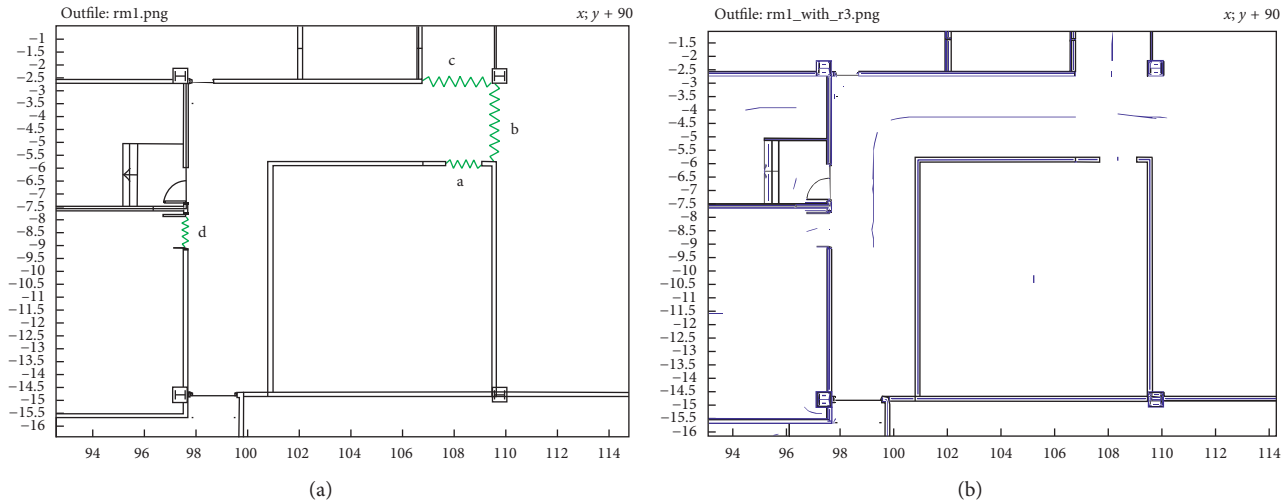



Figure 2: (a) A portion of the building after using heuristics to replace swinging doors with door segments. The zigzag lines a, b, c, and d show where more door segments are needed. (b) The result of adding portions of the line segment Voronoi diagram where the opposition angle is <146°.

also look at the distance from any point on the Voronoi path to its $Q_L$ and $Q_R$ points. Define the *local hallway width* to be twice this distance. It decreases when going from $P_0$ to $P_1$, then it starts increasing and reaches 1.166 times its former value at $P_2$. This temporary decrease in the local corridor width is a good indication that a doorway segment is needed.

More precisely, the requirement for inserting a doorway at Voronoi path point $P_1$ is that the opposition angle there is >157° and

$$W(P_i) - W(P_1) > 0.24 \cdot \sqrt{4\left\|P_i - P_1\right\|^2 + W(P_1)^2} \quad \text{for } i = 0, 2, \tag{1}$$

where $W(\cdot)$ denotes the local corridor width and $P_0$ and $P_2$ are the points on the pruned Voronoi path where the temporary dip in the local corridor width begins and ends.

If the pruned Voronoi path is much shorter than the local corridor width (as it is at the entrance of the large room in Figure 2(b)), this is probably a defect in the Voronoi implementation. Algorithms for the line segment Voronoi diagram are famously difficult to implement reliably, so it is wise to be prepared for such defects. In [12], Held discusses these implementation problems and gives good techniques for minimizing them.

We avoided Held's implementation due to concerns about restrictions against commercial use. The alternative is beyond the scope of this paper—it is a more brute-force approach that asks for a many carefully chosen points $(x, y)$ "which segment's Voronoi region does $(x, y)$ belong to?" In particular, we cope with defects such as the short path in Figure 2(b) by performing "which segment is closest to $(x, y)$?" tests for points $(x, y)$ beyond the ends of the truncated path. This tells what the local corridor width would be at these $(x, y)$ points so that these points can be used as the $P_0$ and $P_2$ points in (1).

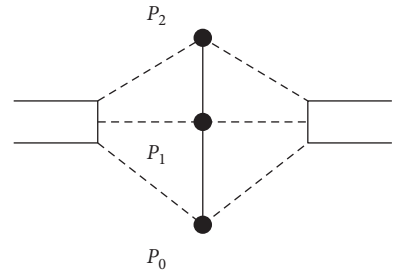



Figure 3: The typical behavior of the pruned line segment Voronoi for a missing doorway opening. The opposition angle increases from 120° to 180° as you go from $P_0$ to $P_1$, then it drops to 120° at $P_2$. The distances from the path to the doorway are 1.166 times as high at $P_0$ and $P_2$ as at $P_1$.

## 3. A Combined Tracking

Although some of the ideas discussed in this paper could apply to a variety of tracking technologies, we focus on RSSI measurements (radio signal strength) for various Wi-Fi access points. For various known locations in the building, a "fingerprint database" provides expected values for a signal-strength vector that gives RSSI versus MAC address for each access point. The dimensionality of the signal-strength vector may be fairly high, for example, 50.

The design goal for the tracking algorithm is to retain the advantages of Kalman filter tracking, while adding room-based analysis. Hence, there is a two-part implementation:

(i) The Kalman filter algorithm (implemented in C++) with minor modifications to facilitate multiple copies of the algorithm running on various alternative rooms

(ii) A supervisor that takes the particles from the Kalman filter and infers likelihoods for the questions "Is it in this room?" and "Is it near a door?"

Section 3.1 briefly outlines the fundamentals of the Kalman filter algorithm and explains how to use smoothed fingerprint measurement data. The Gaussian process smoothing is just one possible source of smoothed data. The approach is general enough to handle any source of RSSI measurements and uncertainty estimates.

Section 3.2 explains how a refined mesh based on Delaunay triangulations can reduce some of the noise in the RSSI values for fingerprint points, while estimating which data points have the most uncertainty. This avoids the $O(n^3)$ behavior of the Gaussian process smoothing and takes walls into account. A wall between two fingerprint database points makes it more reasonable for their RSSI values to differ.

Section 3.3 describes the modifications to make the Kalman filter tracking work well with multiple rooms. Some computations are done on a per-room basis, and particle weights are not normalized separately for each room.

Section 3.4 describes the room chooser and how it interfaces to the Kalman filter algorithm. For the purposes of the following high-level algorithm description, its primary output is a set of alternative rooms with probabilities for each:

(1) Refine the fingerprint data as explained in Section 3.2. This is Algorithm 1.

(2) Initialize the set of rooms to all possible rooms with equal probabilities for each.

(3) Perform a room-based Kalman filter step as explained in Section 3.3.

(4) Use Algorithm 2 from Section 3.4 to update the set of rooms and the probability of being in each room. If not all done, go back to Step (3).

*3.1. A Priori Probabilities for Kalman Filter Tracking.* The object being tracked has various possible trajectories, and we use a set of particles to represent the distribution of these trajectories. Each particle has a position, a velocity, and a weight that represents the relative likelihood for that trajectory. At each time step, we need a quick way to compute a probability for each particle based on how well the observed RSSI values fit the fingerprint data for the particle's position. Call this the *a priori particle probability*. The fundamental idea behind Kalman filter tracking is multiplying this probability into the particle weight and renormalizing the particle weights at each time step. Random Gaussian perturbations to the particle accelerations ensure that the particle cloud covers the entire distribution of possible trajectories.

Let the fingerprint database be

$$\mathscr{D} = \{(L_1, R_1), (L_2, R_2), \ldots, (L_n, R_n)\}, \tag{2}$$

where each $R_j$ is an RSSI vector and $L_j$ is the corresponding location in the building and the $L_j$'s are all different. We need to be able to use $\mathscr{D}$ to compute an a priori particle probability for any location $L$ and any measured RSSI vector $S = s_1, s_2, \ldots, s_k$. In other words, we need to select or interpolate from $\mathscr{D}$ an RSSI vector $R = r_1, r_2, \ldots, r_k$ that is appropriate for $L$, and we need a vector $E = e_1, e_2, \ldots, e_k$ of uncertainty estimates. Treating the components of $E$ as standard deviations gives

$$\sum_{i=1}^{k} \left( \frac{r_i - s_i}{e_i} \right)^2. \tag{3}$$

Then the appropriate way to convert a result $s$ for this sum of squared relative errors into a probability is to use $e^{-s/2}$.

A common way to extend such a fingerprint database $\mathscr{D}$ to more $(x, y)$ locations is via Gaussian process (GP) smoothing. Ferris et al. explained GP in the context of RSSI-based tracking [6]. For our purposes, the key feature of GP is that it can extend (2) to

$$\overline{\mathscr{D}} = \{(\overline{L}_1, \overline{R}_1, \overline{E}_1), (\overline{L}_2, \overline{R}_2, \overline{E}_2), \ldots, (\overline{L}_m, \overline{R}_m, \overline{E}_m)\}, \tag{4}$$

where the locations $\overline{L}_j$ can be chosen to facilitate tracking and each RSSI vector $\overline{R}_j$ is accompanied by a vector of uncertainties $\overline{E}_j$.

For instance, the $\overline{L}_j$ points may be chosen so that they define a regular triangular mesh: tile the building with equilateral triangles and use their vertices as the points $\overline{L}_j$ in (2). In order to evaluate (3) at a point $L$, just compute its barycentric coordinates in its equilateral. Suppose $L = \alpha\overline{L}_{j_1} + \beta\overline{L}_{j_2} + \gamma\overline{L}_{j_3}$, where $\alpha, \beta$, and $\gamma$ add up to 1 and are all nonnegative. Then we can use

$$\begin{aligned} R &= \alpha\overline{R}_{j_1} + \beta\overline{R}_{j_2} + \gamma\overline{R}_{j_3}, \\ E &= \alpha\overline{E}_{j_1} + \beta\overline{E}_{j_2} + \gamma\overline{E}_{j_3}, \end{aligned} \tag{5}$$

in (3).

It is worth noting that any distribution of locations $\overline{L}_j$ in (4) can be used to computing a priori particle probabilities. Each RSSI component (in dB) is a piecewise-linear function of $(x, y)$ if we proceed as follows:

(1) Triangulate the set of locations as well as possible, that is, compute the Delaunay triangulation

(2) Use a data structure that allows rapid identification of the containing triangle for any particle location $L$

(3) Use $L$'s barycentric coordinates in (5) to get $R$ and $E$ for (3)

Unlike the Voronoi diagram for line segments of Section 2.3, Delaunay triangulations of point sets are easy to compute reliably. We used Fortune's implementation of his own algorithm [13] for Step (1).

For Step (2), we used slab decomposition [14] because it is simple and allows fast $O(\log n)$ searches. Its $O(n^2)$ space preprocessing time costs can be avoided by more complicated methods, but we found this unnecessary.

*3.2. Smoother Fingerprint Data on a Refined Mesh.* A major drawback of the Gaussian process approach is that deriving (4) from (2) is very expensive in terms of processing time ($O(mn^3)$ if implemented in the most straightforward manner). Furthermore, it does not allow for any RSSI discontinuities due to interior walls even though radio propagation models generally treat such walls as partly reflective. See the study by Fortune et al. [15] for a discussion of indoor radio propagation.

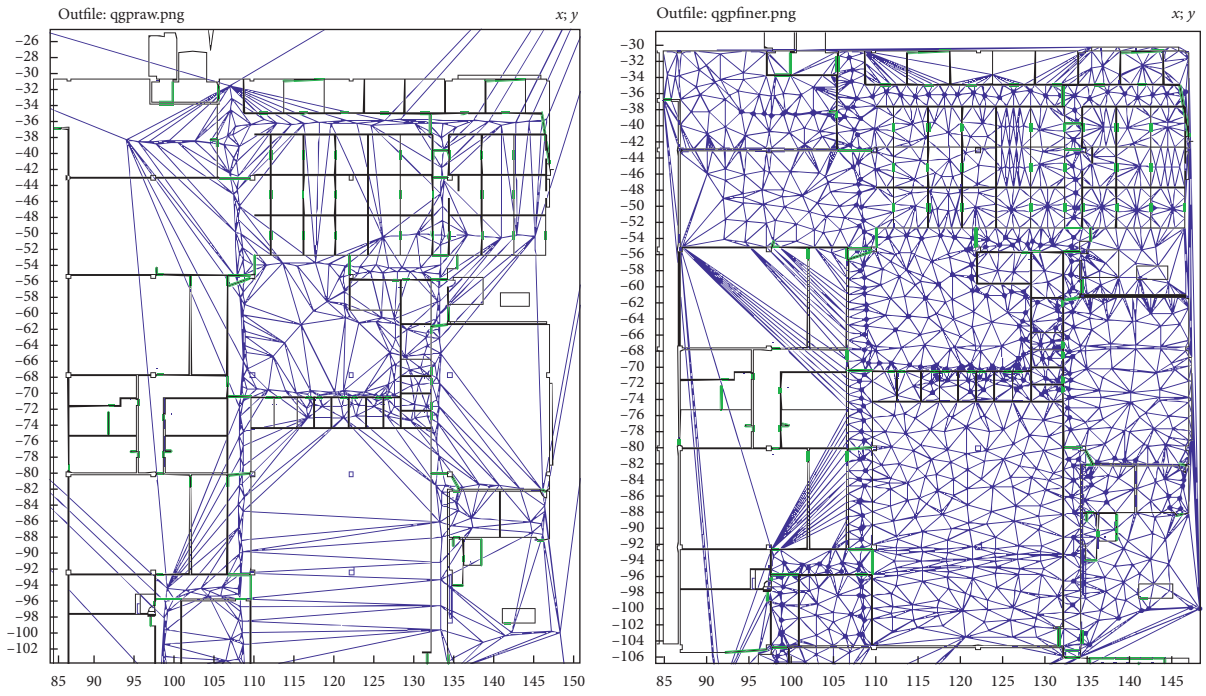



FIGURE 4: (a) The Delaunay triangulation of the unrefined fingerprint database locations with coordinates in meters; (b) a refined version, also Delaunay triangulated, with bold dots at the locations that are also (a) vertices.

As noted in Section 3.1, a priori particle probabilities can be computed directly from the fingerprint database $\mathscr{D}$ if we provide uncertainty vectors $\overline{E}_j$ (perhaps with all entries defaulted to some common value). Figure 4(a) hints at what is wrong with this approach: even with a Delaunay triangulation, many of the triangles are too big and elongated.

The piecewise-linear interpolation is too unsophisticated to be used directly on the large triangles of Figure 4 (a), so we need rules for refining them. These rules that contain heuristic parameters ($H$ with various subscripts) whose values will be discussed in Section 4 are as follows:

(1) Each Delaunay triangle's circumscribing circle must have a radius $\leq H_t$.

(2) No edge in the Delaunay triangulation may cross a wall at more than distance $H_e$ from a wall.

Rule 1 is a natural way to keep the triangles from getting too big because the Delaunay construction guarantees that the circumscribing circle is empty; that is, no vertex of any triangle is strictly within the circle. Rule 2 is needed in order to assign RSSI values to the refined mesh in a manner that takes walls into account. Figure 4(b) shows a refinement that is based on these rules.

Algorithm 1 achieves this refinement by enforcing the two rules. It starts with $n$ measurement point locations $L_j = \overline{L}_j$ that are retained in the refined fingerprint database. These $j \leq n$ have measured RSSI vectors $R_j$ that are ultimately replaced by adjusted vectors $\overline{R}_j$. New vertices $\overline{L}_j$ with $j > n$ are added until we reach some total $m > n$. The adjusted $\overline{R}_j$'s and the $\overline{R}_j$'s for the added vertices are obtained later by solving systems of linear equations in Step (7).

*Algorithm 1.* The fingerprint database refinement algorithm—use $\mathscr{D}$ to compute $\overline{\mathscr{D}}$.

(1) Initialize $m \leftarrow n$, $\overline{L}_i \leftarrow L_i$, and $\overline{R}_i \leftarrow R_i$ for $i \leq n$. $m$ will increase as more points are added. The relation $\overline{L}_i = L_i$ is invariant for all $i \leq n$, but $\overline{R}_1, \overline{R}_2, \ldots, \overline{R}_n$ will change. These changes correct $\overline{R}_1, R_2, \ldots, \overline{R}_n$ to be more consistent with their neighbors.

(2) Add $\overline{L}_j$'s for all the vertices of each room polygon. Sort them by $x$ coordinate, find pairs that are within $\infty$-norm distance $H_t$, and retain only one $\overline{L}_j$ for each such pair.

(3) Find the Delaunay triangulation of all the $\overline{L}_j$'s.

(4) Find the circumscribing circle for each Delaunay triangle and add a new $\overline{L}_j$ for the center of each circle whose radius exceeds $\leq H_t$.

(5) Add new $\overline{L}_j$'s where Delaunay edges cross walls as required by Rule 2.

(6) Go back to Step (3) if any $\overline{L}_j$'s were added in Steps (4) and (5).

(7) For each access point, solve a sparse linear system whose $m$ unknowns are that the access point's component in the $\overline{R}_1, \overline{R}_2, \ldots \overline{R}_m$ vectors. The system sets each $\overline{R}_j$ to a weighted average of $\overline{R}_{j'}$'s for its Delaunay neighbors with weights as explained near

Table 2: Room and doorway scores for one of the room transitions on the "Dublin 87pts forward" track.

| Room 39 | Door 39 → 35 | Door 39 → 70 | Door 39 → 19 | Room 70 |
|---|---|---|---|---|
| 0.9945 | $4 \times 10^{-141}$ | 0.06177 | 0.08559 | 0 |
| 1 | $4\times10^{-19}$ | $5 \times 10^{-09}$ | $9 \times 10^{-06}$ | $9 \times 10^{-09}$ |
| 0.9820 | $3 \times 10^{-12}$ | 0.00099 | 0.0080 | 0 |
| 0.9998 | $9 \times 10^{-45}$ | 0.3088 | 0.3564 | 0.00017 |
| 0.00134 | $2 \times 10^{-51}$ | 0.9666 | 0.518976 | 0.9987 |
| $6 \times 10^{-15}$ | $6 \times 10^{-50}$ | 0.7975 | 0.04788 | 1 |
| $2 \times 10^{-32}$ | 0 | 0 | 0 | 1 |

Each row is one time step (many seconds since the tester repeatedly stopped to gather data).

the end of Section 3.2. If $j \le n$, the $R_j$ counts as a neighbor of $\overline{R}_j$.

(8) For $j = 1, 2, \ldots, m$, set each component of $\overline{E}_j$ to $\sqrt{H_c^2 + (H_s \overline{d}(j))^2 + (H_v \widetilde{e}(j))^2}$ where $\overline{d}(j)$ is the distance from $L_j$ to the closest $L_1, L_2, \ldots, L_n$ point and $\widetilde{e}(j)$ is the $R_i - \overline{R}_i$ for that closest point.

The weights for averaging neighboring RSSI values in Step (7) are basically $1/d$ where $d$ is the distance from $\overline{L}_j$ to the neighboring vertex. If the neighbor is in a different room, the weight is $1/(d + H_p)$. If $j \le n$ and we are treating the $R_j$ measurements as a neighbor of the adjusted values $\overline{R}_j$, the weight is $1/H_m$.

For Step (8), the closest measurement point to a vertex location $\overline{L}_j$ is found via a breadth-first search that adds $H_p$ to the length of any Delaunay edge that crosses from one room to another. If it is convenient to find (say) the closest 4 measurement points $L_{i_1}, \ldots, L_{i+4}$, then $\widetilde{e}(j)$ might be modified to return a weighted average of $R_{i_1} - \overline{R}_{i_1}$ through $R_{i_4} - \overline{R}_{i_4}$ with the weights based on inverse distance.

*3.3. Kalman Filter Tracking for Multiple Rooms.* Recall that Kalman filter tracking multiplies each particle weight by its a priori particle probability (as explained in Section 3.1), and the total weight is renormalized at each time step.

It seems safe to assume that if we know what room to look in, the tracking problem should be easier. This leads to a multiple-room version of the Kalman filter tracking where each particle is constrained to a specific room and the renormalization of total particle weight at each time step is done once for all rooms, not separately for each room being considered. Hence, the total particle weight in a given room is a likelihood estimate for whether the tracked object is in the room:

(i) Each particle has a location $P_i$, a velocity vector $V_i$, a weight $w_i$, and a room number $r_i$.

(ii) The effective particle count $(\sum w_i)^2 / \sum w_i^2$ determines whether particles need to be regenerated, but this decision is made separately for each room. The room's total particle weight is preserved.

(iii) There is a linked list of particles for each active room, and they are freed when the room chooser deselects a room.

*3.4. Using Room and Doorway Scores to Supervise Multiroom Tracking.* The input to the room chooser is a set of particles: the results from Kalman filter tracking at the last time step. The output is a list of room numbers for the next time step, with a suggested location and total particle weight for each selected room. The locations and weights are to be used if the Kalman filter tracker has no suitable left-over information about the room.

The room chooser also needs private data structures to record recent results that are needed for future decisions. For some fixed maximum number of prior time steps such as Nhist = 16, each room has scores ts[ ] and a similar array of *doorway scores* for each door that estimates, for each time step, the conditional probability of being near the door if in the room. Each room also has a "centroid location" for each time step in case the Kalman filter needs to reinitialize for that room and has no existing particles on which to base a position.

The algorithm that uses these data structures for room choosing at one time step involves linear regression. The doorway scores depend on per-room positions that are obtained by averaging particle $P_i$ positions separately for each room. Each doorway score is computed as follows:

(1) Apply linear regression to the distance from the doorway, obtaining a distance estimate for the current time step

(2) Compute a position uncertainty based on the room's variance of $P_i$ positions with added terms proportional to the room dimensions

(3) Use $e^{-\rho^2}$ where $\rho$ is the ratio of distance to uncertainty, but multiply by a term that considers the speed of approach toward the door

Table 2 shows how some of the doorway scores vary as a function of the time step along the "Dublin 87pts forward" test track. All the scores are functions of time, and it is natural to smooth them by applying linear (or logistic) regression. This is trivial because there is a single independent variable. For logistic regression, we transform the probabilities via $y = \ln(p/(1-p))$ then use $p = e^y/(1+e^y)$ to convert the resulting $y$ into a probability. The room choosing algorithm (Algorithm 2) uses (logistic) regression and an array $z[\ ]$ of room weights.

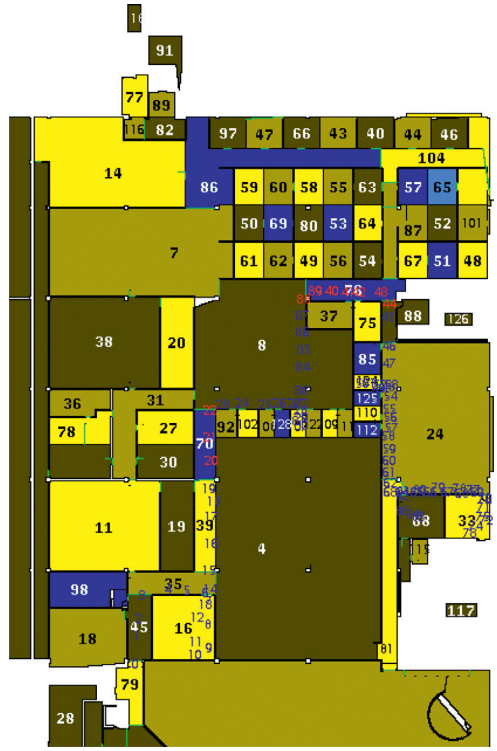



Figure 5: The relevant half of the Dublin building with room polygons shaded to provide a contrast between adjacent rooms. Smaller numbers $1, \ldots, 87$ show the "Dublin 87pts forward" track. ("Dublin 85pts reverse" covers almost the same locations, but in the reverse order.)

*Algorithm 2.* The room choosing algorithm.

(1) Compute doorway scores as explained above and use regression to compute a weight $z[r]$ for each room $r$.

(2) For each doorway, use regression to compute a doorway score $s_{ro}$ for each doorway from a room "$r$" to another room "$o$."

(3) For each doorway score, assign $z[o] = z[o] + s_{ro}z[r] \times (1 - z[o])$. This multiplies $1 - z[o]$ by $1 - s_{ro}z[r]$.

(4) Let $\overline{z}$ be the sum of all room weights $z[r]$.

(5) Sort rooms by descending $z[r]$ and select up to 10 of the high-weight rooms, stopping early if necessary to exclude rooms with $z[r] < \overline{z}/10^4$. (Of course, 10 and $10^4$ could be replaced by other constants.)

## 4. Results

All tests were based on a fingerprint database of 317 locations in the Dublin building as shown in Figure 4(a). Measurements at each location included RSSI values for 51 access points, some of which had multiple MAC addresses. Measurements at each location were repeated 5 times at approximately 1 second intervals so that each RSSI value in the database is the average of at least 5 measurements.

Figure 5 shows how the Dublin building is divided into room polygons There were two test tracks: "Dublin 87pts forward" passes through rooms 45, 35, 16, 35, 39, 70, 8, 107, 8, 76, 88, 24, 124, 81, 24, 33, 24, and 68 and "Dublin 85pts reverse" passes through rooms 81, 68, 24, 112, 24, 85, 88, 76, 87, 104, 86, 14, 7, 8, 108, 8, 111, 8, and 37. The test tracks also had 5 measurements per position; that is, the test subject stopped 87 times to take measurements and recorded the ground truth location. The building actually has a fairly open floor plan, so many of the room boundaries in Figure 5 are partial partitions or places where hallways have 90° bends; for example, rooms 45, 35, 39, and 70 in Figure 5 are all one hallway.

The ground truth locations for the test tracks came close to convex hull of the fingerprint database locations, so it was essential to have a reasonable idea of what the RSSI values should do beyond the convex hull. Simply extrapolating based on Figure 4(b)'s outermost Delaunay triangles is not adequate. We solved this problem by augmenting the fingerprint database with four artificial data points, one at just outside each corner of the building. The artificial points have a common RSSI value $H_q$ for each Wi-Fi access point being measured.

We used "Dublin 87pts forward" track to optimize $H_q$ and the $H_c$, $H_m$, $H_s$, $H_v$, $H_p$, $H_t$, and $H_e$ parameters from Section 3.2. Since the $H_q$ parameter belongs to the tracking program which is considerably faster than the fingerprint refinement algorithm of Section 3.2, we tried 11 values of $H_q$ for each run of the refinement algorithm. We give this 7-dimensional optimization problem to a Nelder-Mead optimizer, and after 588 evaluations, it selected the values in Table 3. The table includes $H_q$ which was hidden from Nelder-Mead and tested exhaustively as explained above (see [15, 16] for a discussion of Nelder-Mead optimization).

Table 3: Heuristic parameter values that optimize "Dublin 87pts forward" results.

| Parameter | Value | Description |
|---|---|---|
| $H_c$ | 10.66 | Constant term for RSSI variance |
| $H_m$ | 0.194 | Distance at which mesh edge is as strong as tie to measurements |
| $H_s$ | 0.166 | RSSI variance term per meter separation from real data point |
| $H_v$ | 1.050 | Weight for neighbor variances in RSSI variance |
| $H_p$ | 10.40 | Distance penalty for edges that cross room boundaries |
| $H_t$ | 2.107 | Refined Delaunay triangles must fit in circle of this radius |
| $H_e$ | 0.411 | Don't allow Delaunay edge across a wall < this from a vertex |
| $H_q$ | −95 | Fingerprint RSSI at artificial corner points |

All distances are in meters.

Table 4: RMS error in meters and room accuracy for various test runs (including the weaker condition that any of the algorithm's top 3 rooms is the correct one).

| Test track | $H_q$ | RMS error (meters) | Top room OK | Among top 3 |
|---|---|---|---|---|
| Dublin 87pts forward | −95 dBm | 3.43 | 64.4% | 95.4% |
| Dublin 85pts reverse | −98 dBm | 6.82 | 54.1% | 76.5% |
| Dublin 36pts reverse | −93 dBm | 3.66 | 55.6% | 91.7% |
| Dublin 85pts reverse | −95 dBm | 12.03 | 29.4% | 51.8% |
| Dublin 36pts reverse | −95 dBm | 12.55 | 44.4% | 55.6% |

One more heuristic parameter $H_f$ was set based on early test runs. This relates to the formula $e^{-s/2}$ for converting the sum $s$ from (3) into an a priori particle probability. To make the probabilities less wild, the actual formula was $e^{-sH_f/2}$, where $H_f = 0.45$.

Table 4 shows tracking accuracy for the test tracks. Results depend strongly on $H_q$, probably because the artificial corner points do not give a very good idea of what RSSI values to expect outside the bounding box of the fingerprint database locations. The "Dublin 36pts reverse" track is the last part of the "Dublin 85pts reverse" track. It was introduced to show that performance can be close to the 3.43 meter optimized value when similar heuristic parameters are used for a different test track.

## 5. Conclusions

A big part of room-based tracking is simply finding the room polygons and constructing a planar subdivision along with the doorways that tell how it is possible to move from room to room. We have seen how the existing approach of extracting walls and doors from CAD drawings can be extended to construct planar subdivisions. The line segment Voronoi diagram is a critical tool for recognizing places where additional doors have to be added. Even if the CAD drawing yields perfect information about doors, the additional doors are needed in order to prevent room polygons from excessively getting large and complicated.

This specific strategy of running Kalman filters in separate rooms is just one of many possible ways to use a room-based planar subdivision. The scheme needs more testing and various engineering improvements to make it more reliable. The reason for retaining multiple guesses as to which room is current is to avoid situations where a wrong choice of starting room cascades to later time steps due to doorway constraints that do not allow travel to the correct room. The results in Section 4 suggest a need for more ways to recover from a bad initial room. Perhaps there should be a restart mechanism.

Another problem that is apparent from Section 4 is that no one value of the corner point RSSI parameter $H_q$ works well in all cases. It is a nontrivial problem to cope with cases where the test track nearly exits the convex hull of the fingerprint measurement locations. The RSSI values at the artificial corner points could be set more intelligently by looking at the fingerprint data near the edge of the convex hull and making separate decisions about each access point. Another option would be to compute the distance to the convex hull and decrease fingerprint RSSI values by some appropriate function of distance. Perhaps the best approach would be to have Section 3.2 solve the problem by extending the refined mesh further outward and adjusting the linear system in Step (7) of Algorithm 1 to handle the extrapolation in some intelligent manner.

Ideally, the measurement points in the fingerprint database should extend far enough beyond the test track so that we do not need a priori particle probabilities beyond the convex hull of the measurement points. Of course it would help to have more fingerprint measurement points, but there are strong incentives to reduce the work required to gather and update such data. Indeed, the purpose of Section 3.2 (or Gaussian process smoothing) is to cope with sparse fingerprint data.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## References


[1] M. Dashti and H. Claussen, "Extracting location information from rf fingerprints," in *Proceedings of 2016 IEEE Globecom Workshops (GC Workshops)*, IEEE, Washington, DC, USA, December 2016.

[2] R. Piché and M. Koivisto, "A method to enforce map constraints in a particle filter's position estimate," in *Proceedings of Positioning, Navigation and Communication (WPNC), 2014 11th Workshop*, pp. 1–4, IEEE, Dresden, Germany, March 2014.

[3] M. Schäfer, C. Knapp, and S. Chakraborty, "Automatic generation of topological indoor maps for real-time map-based localization and tracking," in *Proceedings of Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference*, pp. 1–8, IEEE, Guimaraes, Portugal, September 2011.

[4] M. Klepal, S. Beauregard, and Widyawan, "A novel backtracking particle filter for pattern matching indoor localization," in *Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-Less Environments*, pp. 79–84, ACM, San Francisco, CA, USA, December 2008.

[5] M. Kessel and M. Werner, "Automated WLAN calibration with a backtracking particle filter," in *Proceedings of International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–10, Alcala de Henares, Spain, October 2012.

[6] B. Ferris, D. Haehnel, and D. Fox, "Gaussian processes for signal strength-based location estimation," in *Proceedings of Robotics Science and Systems*, Citeseer, Cambridge, MA, USA, June 2006.

[7] H. Nurminen, A. Ristimaki, S. Ali-Loytty, and R. Piché, "Particle filter and smoother for indoor localization," in *Proceedings of International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–10, Alcala de Henares, Spain, October 2013.

[8] L. Wirola, T. A. Laine, and J. Syrjärinne, "Mass-market requirements for indoor positioning and indoor navigation," in *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference*, pp. 1–7, IEEE, Zürich, Switzerland, 2010.

[9] L. Guibas and J. Stolfi, "Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams," *ACM Transactions on Graphics*, vol. 4, no. 7, pp. 74–123, 1985.

[10] J. D. Hobby, "Practical segment intersection with finite precision output," *Computation Geometry Theory and Applications*, vol. 13, no. 4, pp. 199–214, 1999.

[11] J. D. Hobby, "Generating automatically tuned bitmaps from outlines," *Journal of the ACM*, vol. 40, no. 1, pp. 48–94, 1993.

[12] M. Held, "VRONI: an engineeering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments," *Computational Geometry: Theory and Applications*, vol. 18, no. 2, pp. 95–123, 2001.

[13] S. Fortune, "Sweepline algorithms for Voronoi diagrams," *Algorithmica*, vol. 2, no. 1–4, pp. 153–174, 1987.

[14] D. Dobkin and R. J. Lipton, "Multidimensional searching problems," *SIAM Journal on Computing*, vol. 5, no. 2, pp. 181–186, 1976.

[15] S. J. Fortune, D. M. Gay, B. W. Kernighan, O. Landron, R. A. Valenzuela, and M. H. Wright, "Wise design of indoor wireless systems: practical computation and optimization," *IEEE Computational Science and Engineering*, vol. 2, no. 1, pp. 58–68, 1995.

[16] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 9170746, 18 pages
https://doi.org/10.1155/2017/9170746

*Research Article*

# Positioning Using Terrestrial Multipath Signals and Inertial Sensors

**Christian Gentner, Robert Pöhlmann, Markus Ulmschneider, Thomas Jost, and Siwei Zhang**

*German Aerospace Center (DLR), Institute of Communications and Navigation, Oberpfaffenhofen, 82234 Weßling, Germany*

Correspondence should be addressed to Christian Gentner; christian.gentner@dlr.de

Received 24 February 2017; Revised 15 June 2017; Accepted 20 July 2017; Published 2 October 2017

Academic Editor: Ruizhi Chen



This paper extends an algorithm that exploits multipath propagation for position estimation of mobile receivers named Channel-SLAM. Channel-SLAM treats multipath components (MPCs) as signals from virtual transmitters (VTs) and estimates the positions of the VTs simultaneously with the mobile receiver positions. For Channel-SLAM it is essential to obtain angle of arrival (AoA) measurements for each MPC in order to estimate the VT positions. In this paper, we propose a novel Channel-SLAM implementation based on particle filtering which fuses heading information of an inertial measurement unit (IMU) to omit AoA measurements and to improve the position accuracy. Interpreting all MPCs as signals originated from VTs, Channel-SLAM enables positioning also in non-line-of-sight situations. Furthermore, we propose a method to dynamically adapt the number of particles which significantly reduces the computational complexity. A posterior Cramér-Rao lower bound for Channel-SLAM is derived which incorporates the heading information of the inertial measurement unit (IMU). We evaluate the proposed algorithm based on measurements with a single fixed transmitter and a moving pedestrian carrying the receiver and the IMU. The evaluations show that accurate position estimation is possible without the knowledge of the physical transmitter position by exploiting MPCs and the heading information of an IMU.

## 1. Introduction

Today, most smartphones are equipped with global navigation satellite systems (GNSSs) receivers which allow using applications on the smartphones for navigation [1]. GNSSs provide sufficient position accuracies for mass market application in open sky conditions. However, indoors or in urban canyons the GNSS positioning accuracy could be drastically reduced. In these situations, the GNSS signals might be blocked, degraded by multipath effects, or received with low power. To enhance the positioning performance indoors, different methods and sensor systems can provide position information rather than relying on GNSSs [2–4]. Most of the indoor positioning systems use local infrastructure like positioning with Radio Frequency Identification (RFID) [5], mobile communication base-stations [6, 7], wireless local area network (WLAN) [8], or ultra-wideband (UWB) [9–11]. However, also these wireless radio technologies experience multipath and non-line-of-sight (NLoS) propagation. Multipath propagation is experienced when the transmitted signal arrives at the receiver via several propagation paths. These propagation paths with different delays are caused by reflections, diffractions, and scattering of the electromagnetic wave. Hence, the signal at the receiving antenna consists of a superposition of multiple replicas of the transmitted signal, where each version is called multipath component (MPC) traveling along an individual propagation path. The delay estimate of standard algorithms like the delay locked loop (DLL) is biased in multipath propagation environments [12]. Algorithms like [13–15] reduce the multipath error by modifying the DLL structure. Other algorithms estimate the channel impulse response (CIR) in order to mitigate the influence of multipath propagation on the delay estimate, for example, [16–20]. To retrieve the required delay from the CIR, the path with the smallest delay is treated as the line-of-sight (LoS) path. However, treating the smallest delay as the LoS path may result in weak positioning performance in NLoS

situations. Furthermore, even advanced multipath mitigation algorithms reduce the multipath effects only to a certain degree due to limited signal bandwidth and measurement noise [18].

Nowadays, multipath exploitation instead of mitigation is attracting more and more interest. The authors of [21, 22] exploit multipath propagation for positioning of mobile terminals using multipath fingerprinting algorithms. Other algorithms, for example, [23, 24], interpret reflected signals as signals emitted from virtual transmitters (VTs), where the VT positions are precalculated based on the knowledge of the reflecting surface and physical transmitter positions. Furthermore, the authors of [25] estimate and track the phase information of MPCs using an extended Kalman filter (EKF) and estimate the user position using a time difference of arrival (TDOA) positioning approach. Other algorithms like [26] use a nonlinear least squares algorithm combining UWB measurements at several receiver positions to estimate the positions of the VTs and the receiver simultaneously within small scale scenarios.

This paper describes and extends the multipath assisted positioning algorithm referred to as Channel-SLAM; see [27–31]. Channel-SLAM considers a moving receiver and is suitable for GNSS denied areas like indoor areas. Similarly to other multipath assisted positioning approaches, Channel-SLAM interprets MPCs as LoS signals emitted from VTs. In addition to reflected signals, Channel-SLAM considers also paths occurring due to multiple number of reflections, diffractions, or scattering as well as combinations of these effects. As a consequence, the reception of several MPCs allows position estimation even if only one physical transmitter is present. Interpreting MPCs as directly propagated signals originated from VTs, Channel-SLAM enables positioning also in NLoS situations. Additionally, Channel-SLAM does not require any prior knowledge on locations of reflecting surfaces as Channel-SLAM estimates the receiver position, velocity, clock bias, and the VT positions simultaneously which can be interpreted as simultaneous localization and mapping (SLAM) with radio signals. In [27, 28, 31], we showed that positioning is possible in NLoS scenarios using MPCs without the knowledge of the room geometry by using Channel-SLAM. We investigated in [27] TDOA positioning and especially TDOA between MPCs such that time synchronization between physical transmitters is not essential. In [31], we derived Channel-SLAM based on a Rao-Blackwellized particle filter (RBPF) and compared the accuracy of Channel-SLAM to a derived posterior Cramér-Rao lower bound (PCRLB). However, the Channel-SLAM algorithms in [27, 28, 31] use linear antenna arrays and assume the knowledge of the physical transmitter position.

In this paper, we propose an implementation of Channel-SLAM that uses only a single receiving antenna and fuses similarly to [29, 30] additional information obtained from an inertial measurement unit (IMU). Today many smartphones feature Microelectromechanical System (MEMS) IMUs, which can provide short term relative orientation and position information. Theoretically, the measurements of the IMU can be directly used in an inertial navigation system. However, the position calculation involves double integrations; hence, even small measurement errors quickly cause a drift in the position solution [32]. To avoid that, we only fuse heading measurements from the IMU which solely requires an alignment of the coordinate systems. The heading information of the IMU allows improving the performance of Channel-SLAM by resolving ambiguities and angle of arrival (AoA) measurements are not mandatory anymore. Being a relative positioning system, Channel-SLAM requires an initial prior knowledge of the receiver position and moving direction to define the coordinate system. The positioning algorithm derived in this paper is based on a RBPF where we employ a new transition model for pedestrians. In [29, 30], we showed that positioning with only one physical transmitter is possible if MPCs and heading information from an IMU are used. Compared to [29, 30], the novel transition model enables a performance gain in the position accuracy. In addition to [27–31], we propose a method to dynamically adapt the number of particles which significantly reduces the computational complexity. Furthermore, a PCRLB for Channel-SLAM is derived which incorporates heading information obtained by using an IMU. The developed positioning algorithm is evaluated based on measurement data obtained in an outdoor scenario, where the position of the physical transmitter is unknown. Based on these measurements, we compare the accuracy of Channel-SLAM to that of the derived PCRLB.

The paper is structured as follows: Section 2 describes the signal model; afterwards, Section 3 describes the proposed algorithm which is split into four subsections: Section 3.1 addresses Channel-SLAM; Section 3.2 describes two different transition models using the heading information from an IMU; Section 3.3 summarizes the RBPF; Section 3.4 describes the implementation of the RBPF; afterwards, we derive in Section 4 the PCRLB for Channel-SLAM incorporating the heading changes of the IMU. Thereafter, Section 5 evaluates the algorithm based on measurement data. The last section, Section 6, concludes the paper.

Throughout the paper, we will use the following notations:

(i) $[\cdot]^T$ stands for the vector transpose.

(ii) All vectors are interpreted as column vectors.

(iii) Vectors are denoted by bold small letters.

(iv) $[\mathbf{x}]_l$ denotes the $l$th element of vector $\mathbf{x}$.

(v) $\|\mathbf{A}\|^2 = \sum_l \sum_m |[\mathbf{A}]_{l,m}|^2$ represents the square of the Frobenius norm of $\mathbf{A}$.

(vi) $a \sim \mathcal{N}(\mu_a, \sigma_a^2)$ denotes a Gaussian distributed random variable $a$ with mean $\mu_a$ and variance $\sigma_a^2$.

(vii) $\mathbf{E}[x]$ stands for expectation or sample mean of $x$.

(viii) $1 : k$ stands for all integer numbers starting from 1 to $k$, thus $1, 2, \ldots, k$.

(ix) $\mathsf{p}(x)$ denotes the probability density function of $x$.

(x) $c$ is the speed of light.

(xi) $\hat{x}$ denotes the estimation of $x$.

(xii) $\propto$ stands for proportional.

(xiii) $\{x^{(i)}\}_{i=1}^{N}$ defines the set for $x_i$ with $i = 1, \ldots, N$.

(xiv) $\mathcal{U}[0, N]$ denotes the uniform distribution on the interval $[0, N]$.

## 2. Concept of Virtual Transmitters

Mathematically, the behavior of the multipath channel can be described by the time variant CIR $h(t_k, \tau)$, where $t_k$ indicates the discrete time instants and $\tau$ the delay [33]. According to [33], the CIR $h(t_k, \tau)$ can be assumed to be constant for a short time interval $T$ at discrete time $t_k$ with index $k$,

$$h(t_k, \tau) = \sum_{i=0}^{N(t_k)-1} \alpha_i(t_k) \cdot \delta(\tau - \tau_i(t_k)), \quad (1)$$

for $T_0 \leq t_k \leq T_0 + T$, where $N(t_k)$ is the number of MPCs, $\tau_i(t_k)$ is the delay, $\alpha_i(t_k)$ the complex amplitude of the $i$th MPC, and $\delta(\tau)$ stands for the Dirac distribution [34] (please note that the CIR is generally a summation of an infinite number of MPCs; however, a practical receiver is only capable of capturing signals whose powers are above a certain sensitivity level). For notational conveniences, the LoS propagation path is considered also as a MPC in this paper. Assuming that the transmitted signal $s(t_k)$ is band-limited with bandwidth $B$ and time-limited with a length smaller than $T$, the signal received at time $t_k$ sampled with rate $B$, bin indices $m = 0, \ldots, M-1$, and the delay $\tau_m = m/B$ can be expressed as

$$y(t_k, \tau_m) = \sum_{i=0}^{N(t_k)-1} \alpha_i(t_k)\, s(\tau_m - \tau_i(tk)) + n(\tau_m), \quad (2)$$

where $n(\tau_m)$ denotes the white circular symmetric normal distributed receiver noise with variance $\sigma_n^2$. Using vector notation we obtain from (2)

$$\mathbf{y}(t_k) = [y(t_k, \tau_0), \ldots, y(t_k, \tau_m), \ldots, y(t_k, \tau_{M-1})]. \quad (3)$$

In order to obtain the sparse structure of the CIR from the measurements $\mathbf{y}(t_k)$, super resolution multipath estimation algorithms are necessary. The received signal is geometrically dependent on the transmitter and receiver positions as well as on the environment. Thus, the channel is spatially correlated as long as the spatial sampling is small enough. Hence, we use in this paper the dynamic multipath estimator named Kalman enhanced super resolution tracking (KEST) [20, 35–37] for estimating and tracking multipath parameters. KEST allows estimating the evolution of the CIR over time which is essential for Channel-SLAM as shown in the following section. KEST consists of a Kalman filter (KF) to estimate the complex amplitude $\widehat{\alpha}_i(t_k)$ and delay $\widehat{\tau}_i(t_k)$ for each MPC $i$ utilizing maximum likelihood (ML) estimates as measurements. In the used implementation, KEST uses a standard model for the CIR which comprises a sum of weighted Dirac impulses as in (1). This model describes distinct paths sufficiently well. However, dense multipath components (DMC) lead to a model mismatch in the used KEST implementation. This model mismatch results in an increased variance of the estimated MPC parameters used as measurement noise in Channel-SLAM. For further details about KEST, see [20, 35–37].

To use the delay measurements of the tracked MPCs for positioning, a model describing the delays $\tau_i(t_k)$ depending on the current user position $\mathbf{r}_u(t_k)$ is necessary. For developing such a model, we consider a static environment with a fixed transmitter and a receiver moving along an arbitrary trajectory. Figure 1 summarizes four propagation scenarios; for a detailed description see [31]. In the first scenario, the transmitted signal is reflected on a reflecting surface indicated by the blue lines. For reflection, we consider the effect of an electromagnetic wave reflected by a reflecting surface. When the receiver is moving, the reflection point on the reflecting surface is moving as well. If we mirror the physical transmitter position on the reflecting surface, we obtain the position $\mathbf{r}_{\mathrm{VT},1}$ of $\mathrm{VT}_1$ which is static during the receiver movement. The distance between $\mathrm{VT}_1$ and the receiver is equivalent to the propagation time of the reflected signal multiplied by the speed of light. Hence, the reflected signal can be interpreted as a direct signal from $\mathrm{VT}_1$ to the receiver.

This behavior can be extended to a multiple reflection scenario represented by the red lines. The transmitted signal is reflected two times. Equivalently, the location of $\mathrm{VT}_2$ can be determined by mirroring the transmitter position at both reflecting surfaces, as indicated in Figure 1. The distance between $\mathrm{VT}_2$ and the receiver is equivalent to the propagation time of the reflected signal multiplied by the speed of light. Thus, the signal reflected twice can also be interpreted as a direct signal from $\mathrm{VT}_2$ to the receiver.

Figure 1 exploits by the orange lines additionally a scenario where the signal is scattered, for example, at a lamp post. The propagation effect of scattering occurs if an electromagnetic wave impinges on an object and the energy is spread out in all directions [38]. Geometrically, the effect of scattering can be described as a fixed point $S$ at position $\mathbf{r}_S$ in the pathway of the MPC. We define $S$ as $\mathrm{VT}_3$ at the position $\mathbf{r}_S$ which is constant for all receiver positions for the MPC. Additionally, we treat $d_{\mathrm{VT}} > 0$, the constant distance between physical transmitter and scatterer, as an additional propagation distance associated with the MPC. Hence, the scattered signal can be interpreted as a direct signal from $\mathrm{VT}_3$ to the receiver, however, with a constant offset $d_{\mathrm{VT}}$. Scattering and diffraction can be geometrically described as a fixed point $S$ at position $\mathbf{r}_S$ in the pathway of the MPC and are considered as one model. Hence, unless otherwise stated, the description of scattering is equivalent for diffraction.

The fourth scenario considers the combination of both effects indicated in green. The transmitted signal is scattered at $S$ and afterwards reflected on the first reflecting surface. When the receiver is moving, the reflection point on the reflecting surface is moving as well. Hence, $\mathrm{VT}_4$ is defined by mirroring the scatterer $S$ at the first reflecting surface. Furthermore, between the transmitter and $S$ additional interactions are possible leading to the same position of $\mathrm{VT}_4$.

To summarize, the propagation path of the $i$th MPC can be equivalently described as a direct path with propagation

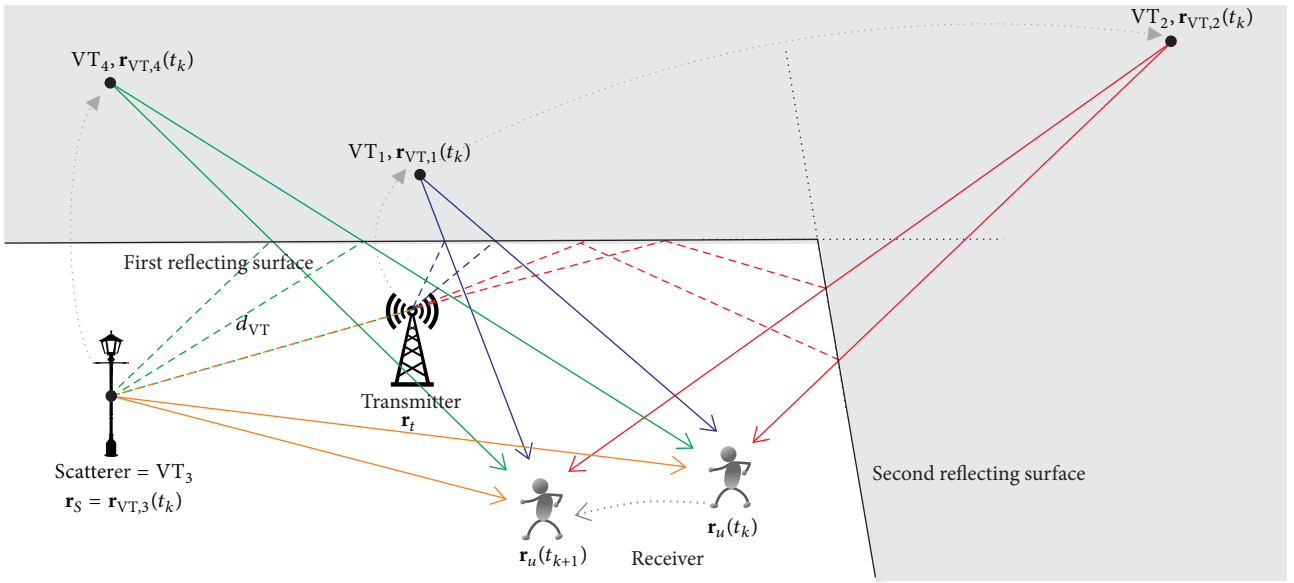



FIGURE 1: The figure shows four propagation scenarios: First scenario (blue): the transmitted signal is reflected on a reflecting surface. $VT_1$ is defined by mirroring the physical transmitter position at the surface. Second scenario (red): the transmitted signal is reflected twice. $VT_2$ is defined by mirroring the physical transmitter position at both surfaces. Third scenario (orange): the transmitted signal is scattered at $S$. $VT_3$ is defined at the position of $S$. Fourth scenario (green): the transmitted signal is scattered and afterwards reflected on a reflecting surface. $VT_4$ is defined by mirroring the scatterer $S$ at the surface.

length $d_i(t_k)$ between $VT_i$ and the receiver plus an additional constant propagation length $d_{VT,i}(t_k)$; hence,

$$\begin{aligned} d_i(t_k) &= \tau_i(t_k) \cdot c \\ &= \left\| \mathbf{r}_u(t_k) - \mathbf{r}_{VT,i}(t_k) \right\| + d_{VT,i}(t_k), \end{aligned} \tag{4}$$

where $c$ denotes the speed of light and $\mathbf{r}_{VT,i}(t_k)$ the position of the $i$th VT (please note that the position of the VTs and the additional propagation lengths are constant over time. Nevertheless for notational convenience a time dependence on $t_k$ is introduced here). The additional propagation length is zero, that is, $d_{VT,i}(t_k) = 0$, if only reflections occurred on the pathway between physical transmitter and receiver or greater than zero, that is, $d_{VT,i}(t_k) > 0$, if the MPC is interacting with at least one scatterer. In general, $d_{VT,i}(t_k)/c$ can be interpreted as a clock offset between the $i$th VT and the physical transmitter.

## 3. Channel-SLAM

*3.1. Position Estimation.* Figure 2 presents the available sensors together with the corresponding measurements. As shown on the left, we measure the sampled received signal $\mathbf{y}(t_k)$ as stated in (3) where we assume that the transmitter continuously emits known wideband signals. Based on $\mathbf{y}(t_k)$, the multipath parameters amplitude $\alpha_i(t_k)$ and delay $\tau_i(t_k) = d_i(t_k)/c$ for each MPC are estimated and tracked by KEST. The estimated propagation path lengths $\hat{d}_i(t_k) = \hat{\tau}_i(t_k) \cdot c$ of all $N(t_k)$ MPCs of KEST are used as measurements

$$\mathbf{z}(t_k) = \left[ \hat{d}_0(t_k), \ldots, \hat{d}_{N(t_k)-1}(t_k) \right]^T \tag{5}$$

in Channel-SLAM with the corresponding variances $\boldsymbol{\sigma}_z(t_k)$. Because the VT positions are unknown, the receiver position and the positions of the VTs have to be estimated simultaneously. Thus, the state vector $\mathbf{x}(t_k)$ describing the complete system at time instant $t_k$ for $N(t_k)$ MPCs is

$$\mathbf{x}(t_k) = \left[ \mathbf{x}_u(t_k)^T, \mathbf{x}_{VT}(t_k)^T \right]^T, \tag{6}$$

with the receiver states $\mathbf{x}_u(t_k)$ and the VT states $\mathbf{x}_{VT}(t_k)$. The receiver state $\mathbf{x}_u(t_k)$ includes the receiver position $\mathbf{r}_u(t_k)$, the receiver velocity $\mathbf{v}_u(t_k)$, and the receiver's clock bias $b_u(t_k)$; hence,

$$\mathbf{x}_u(t_k) = \left[ \mathbf{r}_u(t_k)^T, \mathbf{v}_u(t_k)^T, b_u(t_k) \right]^T. \tag{7}$$

According to the description given in the previous section and (4), an MPC can be represented by a direct path between a VT and the receiver plus an additional propagation length. Hence, the parameters representing the $i$th VT are defined as

$$\mathbf{x}_{VT,i}(t_k) = \left[ \mathbf{r}_{VT,i}(t_k)^T, d_{VT,i}(t_k) \right]^T, \tag{8}$$

where $\mathbf{r}_{VT,i}(t_k)$ is the position of the $i$th VT and $d_{VT,i}(t_k)$ the additional propagation length. Using vector notation for all VTs, we obtain

$$\mathbf{x}_{VT}(t_k) = \left[ \mathbf{x}_{VT,0}(t_k)^T, \ldots, \mathbf{x}_{VT,N(t_k)-1}(t_k)^T \right]^T. \tag{9}$$

Additionally, as illustrated in Figure 2, an IMU is used. The IMU provides measurements of the acceleration $\mathbf{a}^b(t_k)$ and turn rates $\boldsymbol{\omega}_{ib}^b(t_k)$ in three dimensions. After calibration,

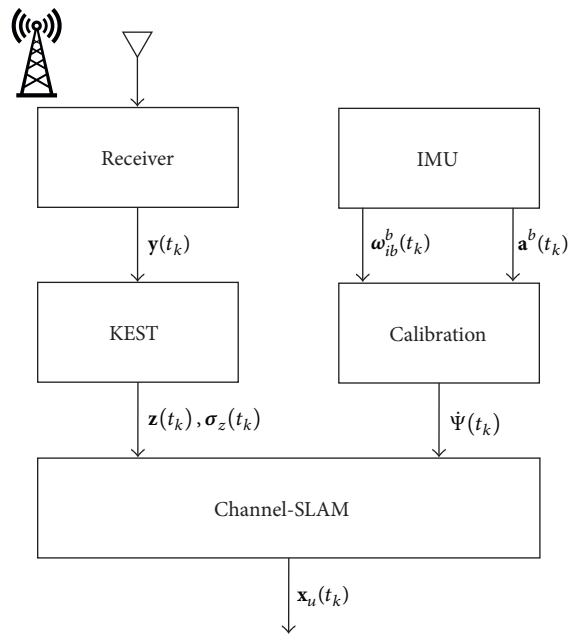



Figure 2: System model consisting of a terrestrial receiver and an IMU.

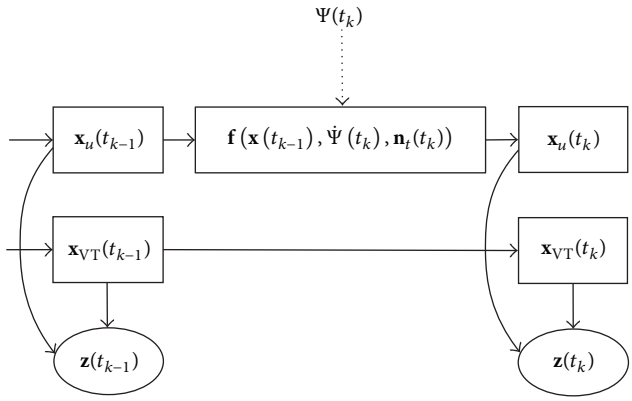



Figure 3: First-order hidden Markov model representing the dynamic system of Channel-SLAM.

the heading change $\dot{\Psi}(t_k)$ is used in Channel-SLAM as a control input and is therefore directly integrated into the transition model.

We use a discrete time representation for the transition and measurement model of the dynamic system with

$$\mathbf{x}(t_k) = \mathbf{f}\left(\mathbf{x}(t_{k-1}), \dot{\Psi}(t_k), \mathbf{n}_t(t_k)\right), \tag{10}$$

$$\mathbf{z}(t_k) = \mathbf{h}(\mathbf{x}(t_k), \mathbf{n}_h(t_k)). \tag{11}$$

The transition model in (10) describes the state evolution from time instant $t_{k-1}$ to time instant $t_k$ employing a possible nonlinear function $\mathbf{f}(\cdot,\cdot,\cdot)$ with the process noise $\mathbf{n}_t(t_k)$ and using a control input which is in our case the heading change $\dot{\Psi}(t_k)$. The control input is considered as perfectly known and hence error-free. The measurement model (11) relates the state vector to the measurements by a possible nonlinear function $\mathbf{h}(\cdot,\cdot)$ and the measurement noise $\mathbf{n}_h(t_k)$ at time instant $t_k$. Figure 3 shows the considered dynamic Bayesian network, that is, a first-order hidden Markov model.

Equations (10) and (11) can also be interpreted from a Bayesian perspective: based on measurements, we want to recursively estimate the unknown probability density function (PDF) of the state $\mathbf{x}(t_k)$. In a recursive Bayesian formulation, this problem can be described as finding the posterior probability distribution

$$\mathrm{p}\left(\mathbf{x}(t_k) \mid \mathbf{z}(t_{1:k}), \dot{\Psi}(t_{1:k}), \mathbf{x}(t_0)\right). \tag{12}$$

Recursive Bayesian filtering provides a methodology to optimally estimate (12) by a prediction step to calculate $\mathrm{p}(\mathbf{x}(t_k) \mid \mathbf{z}(t_{1:k-1}), \dot{\Psi}(t_{1:k}), \mathbf{x}(t_0))$ and an update step to obtain $\mathrm{p}(\mathbf{x}(t_k) \mid \mathbf{z}(t_{1:k}), \dot{\Psi}(t_{1:k}), \mathbf{x}(t_0))$ which considers the measurement $\mathbf{z}(t_k)$ at time instant $t_k$ with the likelihood function $\mathrm{p}(\mathbf{z}(t_k) \mid \mathbf{x}(t_k))$ [39, 40]. By assuming independence between the transition priors of the receiver state vector $\mathbf{x}_u(t_k)$ and the VT state

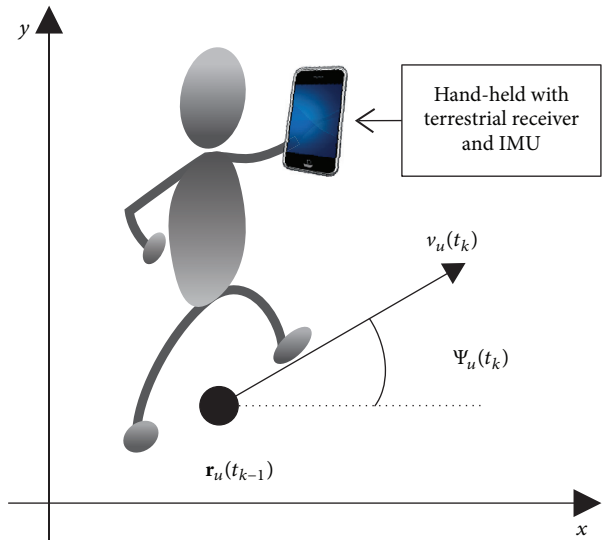



Figure 4: Illustration of the prediction model for the pedestrian.

vectors $\mathbf{x}_{\mathrm{VT},i}(t_k)$ associated with the MPCs $i = 0, \ldots, N(t_k)-1$, the transition prior $\mathrm{p}(\mathbf{x}(t_k) \mid \mathbf{x}(t_{k-1}), \dot{\Psi}(t_k), \mathbf{x}(t_0))$ is defined here as

$$\begin{aligned}&\mathrm{p}\left(\mathbf{x}\left(t_k\right) \mid \mathbf{x}\left(t_{k-1}\right), \dot{\Psi}\left(t_k\right), \mathbf{x}\left(t_0\right)\right)\\ &\quad = \mathrm{p}\left(\mathbf{x}_u\left(t_k\right) \mid \mathbf{x}_u\left(t_{k-1}\right), \dot{\Psi}\left(t_k\right), \mathbf{x}_u\left(t_0\right)\right)\\ &\qquad \times \prod_{i=0}^{N(t_k)-1} \mathrm{p}\left(\mathbf{x}_{\mathrm{VT},i}\left(t_k\right) \mid \mathbf{x}_{\mathrm{VT},i}\left(t_{k-1}\right)\right),\end{aligned} \tag{13}$$

where we inherently assume independence among MPCs, that is, propagation paths interacting with distinct objects. This is based on the well-known uncorrelated scattering assumption in wireless propagation channel modelling [38]. We obtain for the transition prior $\mathrm{p}(\mathbf{x}_{\mathrm{VT},i}(t_k) \mid \mathbf{x}_{\mathrm{VT},i}(t_{k-1}))$ of the $i$th MPC

$$\begin{aligned}&\mathrm{p}\left(\mathbf{x}_{\mathrm{VT},i}\left(t_k\right) \mid \mathbf{x}_{\mathrm{VT},i}\left(t_{k-1}\right)\right)\\ &\quad = \delta\left(\mathbf{x}_{\mathrm{VT},i}\left(t_k\right) - \mathbf{x}_{\mathrm{VT},i}\left(t_{k-1}\right)\right).\end{aligned} \tag{14}$$

For the transition prior $\mathrm{p}(\mathbf{x}_u(t_k) \mid \mathbf{x}_u(t_{k-1}), \dot{\Psi}(t_k))$ of the receiver state vector we provide in Section 3.2 two models indicated by the function $\mathbf{f}(\mathbf{x}_u(t_{k-1}), \dot{\Psi}(t_k), \mathbf{n}_u(t_k))$ in Figure 3.

Assuming the elements of $\mathbf{z}(t_k)$ to be independent Gaussian distributed conditioned on the current state $\mathbf{x}(t_k)$, $\mathrm{p}(\mathbf{z}(t_k) \mid \mathbf{x}(t_k))$ can be expressed as

$$\begin{aligned}&\mathrm{p}\left(\mathbf{z}\left(t_k\right) \mid \mathbf{x}\left(t_k\right)\right)\\ &\quad = \prod_{i=0}^{N(t_k)-1} \frac{1}{\sqrt{2\pi}\sigma_{d,i}\left(t_k\right)} e^{-(\widehat{d}_i(t_k)-d_i(t_k))^2/2\sigma^2_{d,i}(t_k)}\end{aligned} \tag{15}$$

with the propagation length

$$d_i\left(t_k\right) = \left\|\mathbf{r}_u\left(t_k\right) - \mathbf{r}_{\mathrm{VT},i}\left(t_k\right)\right\| + d_{\mathrm{VT},i}\left(t_k\right) + b_u\left(t_k\right) \cdot c, \tag{16}$$

for the $i$th MPC, where $\sigma^2_{d,i}(t_k)$ denotes the corresponding variances.

*3.2. Prediction Model Using Heading Changes.* This paper considers a moving pedestrian carrying a hand-held device equipped with a terrestrial receiver and an IMU. A variety of pedestrian transition models exist in literature, for example, [41–44]; however, they do not fit for the considered application. Many of them focus on movements of groups, use additional information like floor plans, or do not incorporate information from an IMU. IMUs include in general accelerometers measuring acceleration $\mathbf{a}^b(t_k)$ and gyroscopes measuring turn rates $\boldsymbol{\omega}^b_{ib}(t_k)$, as indicated in Figure 2. These measurements are provided with respect to the sensor alignment [32], that is, the body frame. In order to obtain the measurements in a two-dimensional Cartesian coordinate system as shown in Figure 4, a transformation between the coordinate systems is necessary; see, for example, [45]. In our considered measurement scenario, the position of the IMU is assumed as constant with respect to the receiving antenna. Therefore, we are able to calculate the coordinate transformation matrices during a calibration phase when the pedestrian is standing still at the beginning. For other systems, where the sensor is decoupled, the sensor orientation has to be estimated continuously by applying strapdown navigation together with in-field calibration [46].

We propose two different constant velocity models, a linear model with Gaussian noise and a nonlinear model with Rician noise.

*3.2.1. Gaussian-Transition-Model.* The first proposed transition model is based on a discrete white noise acceleration model [47], referred to as Gaussian-Transition-Model, with

$$\overline{\mathbf{x}}_u\left(t_k\right) = \mathbf{A}_u\left(t_\delta, \dot{\Psi}\left(t_k\right)\right)\overline{\mathbf{x}}_u\left(t_{k-1}\right) + \mathbf{n}_t\left(t_k\right), \tag{17}$$

in a two-dimensional Cartesian coordinate system. The receiver state vector $\bar{\mathbf{x}}_u(t_k) = [\bar{\mathbf{r}}_u(t_k), \bar{\mathbf{v}}_u(t_k), b_u(t_k)]^T$ consists of the $x$-$y$ positions

$$\bar{\mathbf{r}}_u(t_k) = \left[r_{u,x}(t_k), r_{u,y}(t_k)\right]^T \tag{18}$$

and the velocities

$$\bar{\mathbf{v}}_u(t_k) = \left[v_{u,x}(t_k), v_{u,y}(t_k)\right]^T, \tag{19}$$

where $v_{u,x}(t_k), v_{u,y}(t_k)$ are the corresponding velocities in $x$-$y$ direction and the receiver's clock bias $b_u(t_k)$ where a standard clock bias model is used [12, 48] (Please note that if transmitter and receiver oscillators provide different frequencies, a clock drift parameter has to be considered additionally). The transition matrix $\mathbf{A}_u(t_\delta, \dot{\Psi}(t_k))$ in (17) includes a rotation matrix with the heading changes $\dot{\Psi}(t_k)$, with

$$\mathbf{A}_u\left(t_\delta, \dot{\Psi}(t_k)\right) = \begin{pmatrix} 1 & 0 & t_\delta & 0 & 0 \\ 0 & 1 & 0 & t_\delta & 0 \\ 0 & 0 & \cos\left(\dot{\Psi}(t_k)\right) & -\sin\left(\dot{\Psi}(t_k)\right) & 0 \\ 0 & 0 & \sin\left(\dot{\Psi}(t_k)\right) & \cos\left(\dot{\Psi}(t_k)\right) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{20}$$

where $t_\delta = t_k - t_{k-1}$ and $\mathbf{n}_t(t_k) \sim \mathcal{N}(0, \mathbf{Q}_u(t_\delta))$ is the transition noise of the receiver state vector with covariance

$$\mathbf{Q}_u(t_\delta) = \begin{pmatrix} \sigma_{q_u}^2 \frac{(t_\delta)^3}{3} & 0 & \sigma_{q_u}^2 \frac{(t_\delta)^2}{2} & 0 & 0 \\ 0 & \sigma_{q_u}^2 \frac{(t_\delta)^3}{3} & 0 & \sigma_{q_u}^2 \frac{(t_\delta)^2}{2} & 0 \\ \sigma_{q_u}^2 \frac{(t_\delta)^2}{2} & 0 & \sigma_{q_u}^2 t_\delta & 0 & 0 \\ 0 & \sigma_{q_u}^2 \frac{(t_\delta)^2}{2} & 0 & \sigma_{q_u}^2 t_\delta & 0 \\ 0 & 0 & 0 & 0 & \sigma_{q_b}^2 \end{pmatrix}, \tag{21}$$

where $\sigma_{q_u}^2$ defines the continuous-time process noise intensity that has to be set based on the application with physical dimension $[\mathrm{m}^2/\mathrm{s}^3]$ and $\sigma_{q_b}^2$ the variance of the clock bias. This transition model is similar to the transition model presented in [29, 30], with the advantage that the transition model is linear if the heading changes are known.

Since we incorporate only the heading changes $\dot{\Psi}(t_k)$, we do not have any speed measurements and the speed has to be estimated implicitly. In order to adapt quickly to different walking speeds, $\sigma_{q_u}$ has to be large or many particles have to be used to cover all possible movements. A large value for $\sigma_{q_u}$ may cause backward movements of the transition model which results in estimation errors of Channel-SLAM. Another drawback of this transition model is that the estimated state information is completely lost when the user is standing; thus, the velocity components are zero. In order to overcome the mentioned problems, we develop a second transition model as described in the following.

*3.2.2. Rician-Transition-Model.* Similarly to Section 3.2.1, we follow a two-dimensional positioning approach in the Cartesian coordinate system with the receiver state vector $\bar{\mathbf{x}}_u(t_k) = [\check{\mathbf{r}}_u(t_k), \check{\mathbf{v}}_u(t_k), b_u(t_k)]^T$ which consists of the $x$-$y$ receiver positions $\check{\mathbf{r}}_u(t_k)$ as defined in (18), the receiver velocity vector $\check{\mathbf{v}}_u(t_k)$, and the receiver's clock bias $b_u(t_k)$. The receiver velocity vector is

$$\check{\mathbf{v}}_u(t_k) = \left[v_u(t_k), \Psi_u(t_k)\right]^T, \tag{22}$$

where $v_u(t_k)$ is the receiver speed and $\Psi_u(t_k)$ the heading of the receiver; see Figure 4. The heading $\Psi_u(t_k)$ describes the walking direction of the pedestrian with respect to the Cartesian coordinate system. Hence, we can define the transition model with

$$\check{\mathbf{r}}_u(t_k) = \check{\mathbf{r}}_u(t_{k-1}) + t_\delta v_u(t_k) \begin{bmatrix} \cos\left(\Psi_u(t_k)\right) \\ \sin\left(\Psi_u(t_k)\right) \end{bmatrix}, \tag{23}$$

where the velocity follows a Rician distribution with

$$v_u(t_k) \sim \mathcal{R}\left(v_u(t_{k-1}), \sigma_m(t_k)\right), \tag{24}$$

with scale parameter $\sigma_m(t_k)$. For speeds close to zero, the Rician distribution approximates a Rayleigh distribution; thus, the speed is always positive. Hence, it has the advantage of preventing the filter from converging to negative velocities, which are highly unlikely regarding a normal pedestrian walking behavior (Please note that the developed transition model does not include standing or walking backwards phases. This could be additionally considered by extracting more information from the IMU measurements). This is important for our approach, since ambiguities of Channel-SLAM could otherwise cause a movement in the wrong direction. On the other hand, for higher speeds, the distribution becomes approximately Gaussian which reflects empirical data of pedestrian walking speeds [43].

Finally, the heading of the user is defined by

$$\Psi_u(t_k) = \Psi_u(t_{k-1}) + (t_k - t_{k-1})\dot{\Psi}(t_k) + w_\Psi(t_k), \tag{25}$$

where $\dot{\Psi}(t_k)$ is the heading change from the IMU after calibration with the heading noise $w_\Psi(t_k)$ using a von Mises distribution. For the transition prior of the clock bias we use similar to Section 3.2.1 a standard clock bias model $b_u(t_k) = b_u(t_{k-1}) + n_b(t_{k-1})$, where $n_b(t_{k-1})$ defines the transition noise with the variance $\sigma_{q_b}^2$ [12, 48] (Please note that if transmitter and receiver oscillators provide different frequencies, a clock drift parameter has to be considered additionally). In the following we refer to this transition model as Rician-Transition-Model.

*3.3. Rao-Blackwellized Particle Filter.* As introduced in [31], Channel-SLAM is derived based on Rao-Blackwellization where the state space of $\mathbf{x}(t_k)$ is partitioned into subspaces.

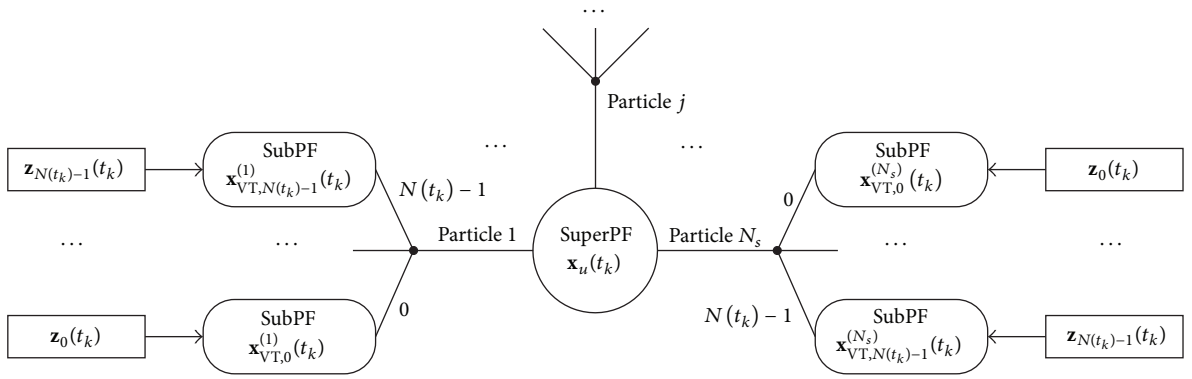



FIGURE 5: The algorithm is based on a superordinate particle filter (superPF) and subordinate particle filters (subPFs). Each particle $j = 1 \cdots N_s$ of the superPF consists of $N(t_k)$ subPFs.

Hence, we use particle filters (PFs) to estimate the subspaces representing the VTs inside a PF. The reason to use a PF instead of a low complexity EKF is the high nonlinearity of the measurements in (16). As shown in Figure 5, the algorithm is based on a superordinate particle filter (superPF) and subordinate particle filters (subPFs): Each particle $j = 1 \cdots N_s$ of the superPF with the state vector $\mathbf{x}_u^{(j)}(t_k) = [\mathbf{r}_u^{(j)}(t_k)^T, \mathbf{v}_u^{(j)}(t_k)^T, b_u^{(j)}(t_k)]^T$ consists of $N(t_k)$ subPFs. Each subPF is represented by the particles $\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k)$ with $a = 1, \ldots, N_{P,j,i}(t_k)$, where $N_{P,j,i}(t_k)$ stands for the number of particles in the $i$th subPF with $i = 0, \ldots, N(t_k)-1$, estimating $\mathbf{x}_{\mathrm{VT},i}^{(j)}(t_k)$. Using subPFs for each VT allows using different numbers of particles in each subPF and, furthermore, allows a dynamic adjustment of the number of particles for each subPF introduced in Section 3.4.

According to [31], the marginalized posterior filtered density $\mathrm{p}(\mathbf{x}_u(t_k) \mid \mathbf{z}(t_{1:k}), \dot{\Psi}(t_{1:k}))$ of the superPF can be approximated by importance samples (see [31, 39]) as

$$\mathrm{p}\left(\mathbf{x}_u\left(t_k\right) \mid \mathbf{z}\left(t_{1:k}\right), \dot{\Psi}\left(t_{1:k}\right)\right) \approx \sum_{j=1}^{N_s} w^{(j)}\left(t_k\right) \delta\left(\mathbf{x}_u\left(t_k\right) - \mathbf{x}_u^{(j)}\left(t_k\right)\right), \tag{26}$$

where $w^{(j)}(t_k)$ defines the weight for the $j$th particle at time instant $t_k$ with

$$w^{(j)}\left(t_k\right) \propto \mathrm{p}\left(\mathbf{z}\left(t_k\right) \mid \mathbf{x}_u^{(j)}\left(t_k\right), \mathbf{z}\left(t_{k-1}\right)\right) \propto \prod_{i=0}^{N(t)-1} \sum_{a=1}^{N_{P,i,j}(t_k)} w_i^{(j,a)}\left(t_k\right) \tag{27}$$

and the weight $w_i^{(j,a)}(t_k)$ of the subPFs at time instant $t_k$ with

$$w_i^{(j,a)}\left(t_k\right) \triangleq \mathrm{p}\left(\widehat{d}_i\left(t_k\right) \mid \mathbf{x}_u^{(j)}\left(t_k\right), \mathbf{x}_{\mathrm{VT},i}^{(j,a)}\left(t_k\right)\right). \tag{28}$$

Contrarily to [31], resampling is performed at each time instant to prevent degeneration; hence, (27) and (28) do not depend on the weights $w^{(j)}(t_{k-1})$ and $w_i^{(j,a)}(t_{k-1})$. Additionally, the derivations in [31] consider a regularized PF [39] where $\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k)$ is drawn after resampling from the Gaussian-Kernel $K(\cdot)$. The Gaussian-Kernel $K(\cdot)$ improves the robustness of Channel-SLAM to cope with small model mismatches in the measurements.

*3.4. Particle Filter Implementation.* Algorithm 1 provides the pseudocode of Channel-SLAM, which is executed at every time instant $t_k \geq t_0$ with the estimates $\mathbf{z}(t_k), \boldsymbol{\sigma}_z(t_k)$ obtained from KEST. During the initialization, at time instant $t_k = t_0$, the particles $\{\mathbf{x}_u^{(j)}(t_0)\}_{j=1}^{N_s}$ of the superPF are initialized according to prior knowledge. The particles $\{\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_0)\}_{a=1}^{N_{p,j,i}}$ of the subPFs are initialized dependent on $\mathbf{x}_u^{(j)}(t_0)$ and the measurements $\widehat{d}_i(t_0)$ for the $i$th MPC. To initialize the states of $\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_0)$ with $a = 1, \ldots, N_{p,j,i}$ of the $j$th subPF associated with the $i$th MPC a grid is used. The positions $\mathbf{r}_{\mathrm{VT},i}^{(j,a)}(t_0)$ of the particles $\{\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_0)\}_{a=1}^{N_{P,j,i}}$ are distributed on a grid inside a circle around $\mathbf{r}_u^{(j)}(t_0)$ with radius $\widehat{d}_i(t_0) + \Delta_d$ such that

$$0 \leq \left\| \mathbf{r}_{\mathrm{VT},i}^{(j,a)}\left(t_0\right) - \mathbf{r}_u^{(j)}\left(t_0\right) \right\| \leq \widehat{d}_i\left(t_0\right) + \Delta_d \tag{29}$$

with spacing $\Delta_d$ (the number of grid points $N_{P,j,i}(t_k)$ can be estimated by Gauss's circle problem). The additional propagation length is $d_{\mathrm{VT},i}^{(j,a)}(t_0) = \widehat{d}_i(t_0) - \|\mathbf{r}_{\mathrm{VT},i}^{(j,a)}(t_0) - \mathbf{r}_u^{(j)}(t_0)\|$, where we inherently assume $b_u(t_0) = 0$ for the initialization. Hence, the total number of particles can be calculated as

$$N_t\left(t_k\right) = \sum_{j=1}^{N_p} \sum_{i=0}^{N(t_k)-1} N_{P,j,i}\left(t_k\right). \tag{30}$$

For each particle $j$ of the superPF, the receiver state $\mathbf{x}_u^{(j)}(t_{k-1})$ is propagated according to the transition model described in Section 3.2 indicated by the function $\mathbf{f}(\mathbf{x}_u^{(j)}(t_{k-1}), \dot{\Psi}(t_k), \mathbf{n}_t(t_k))$ using the heading changes $\dot{\Psi}(t_k)$ (cf. Line (5) in Algorithm 1). Afterwards, Channel-SLAM determines whether the number of tracked MPCs has changed. In case that new MPCs have been detected, new subPFs are added and initialized using (29) (cf. Line (7) in Algorithm 1). In case that MPCs are not tracked by KEST

**Input**:
Multipath estimates: $\mathbf{z}(t_k)$, $\boldsymbol{\sigma}_z(t_k)$ and $\dot{\Psi}(t_k)$;
States for $t_k > t_0$:
$$\left\{\mathbf{x}_u^{(j)}(t_{k-1}), \left\{\left\{\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_{k-1})\right\}_{a=1}^{N_{p,j,i}}\right\}_{i=0}^{N(t_{k-1})-1}\right\}_{j=1}^{N_s}$$
**Output**:
States for $\geq t_0$:
$$\left\{\mathbf{x}_u^{(j)}(t_k), \left\{\left\{\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k)\right\}_{a=1}^{N_{p,j,i}}\right\}_{i=0}^{N(t_k)-1}\right\}_{j=1}^{N_s}$$
MMSE estimate: $\hat{\mathbf{x}}(t_k)$ for $t_k > t_0$

(1) **if** $t_k = t_0$ **then**
(2) Initialization using $\mathbf{z}(t_k)$ and $\boldsymbol{\sigma}_z(t_k)$;
(3) **else**
(4) **for** $j = 1 : N_s$ **do**
(5) Draw $\mathbf{x}_u^{(j)}(t_k) = \mathbf{f}(\mathbf{x}_u^{(j)}(t_{k-1}), \dot{\Psi}(t_k), \mathbf{w}(t_{k-1}))$;
(6) **if** *New paths detected* **then**
(7) Initialize new subPFs;
(8) **if** *Tracking of paths lost* **then**
(9) Delete corresponding subPFs;
(10) **for** $i = 0 : N(t_k) - 1$ **do**
(11) **for** $a = 1 : N_{P,j,i}(t_k)$ **do**
(12) Assign $\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k) = \mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_{k-1})$;
(13) Calculate $w_i^{(j,a)}(t_k) = \mathrm{p}(\mathbf{z}_i(t_k) \mid \mathbf{x}_u^{(j)}(t_k), \mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k))$;
(14) Calculate total subPF weight:
$t_{j,i} = \mathrm{SUM}\left[\{w_i^{(j,a)}(t_k)\}_{a=1}^{N_{p,j,i}(t_k)}\right]$;
(15) $w^{(j)}(t_k) = \prod_{i=0}^{N(t_k)-1} t_{j,i}$;
(16) Resample using Algorithm 2;
(17) Calculate MMSE $\hat{\mathbf{x}}(t_k)$ according to (31);

Algorithm 1: Channel-SLAM for time instant $t_k$.

anymore, the corresponding subPFs are removed (cf. Line (9) in Algorithm 1). Equivalent to [31], neither KEST nor Channel-SLAM considers retracking of previous MPCs. Hence, if the tracking of an MPC has been lost and is regained again, the corresponding VT is initialized without any prior information. According to (14), the state $\mathbf{x}_{\mathrm{VT},i}(t_k)$ is time-invariant; hence, each subPF assigns the states of the VTs with $\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k) = \mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_{k-1})$. Thereafter, the weights for the subPFs and superPF are calculated using (28) and (27).

Afterwards, the subPFs and superPF are resampled. The basic idea of the resampling method is to eliminate particles with low weights and reproduce particles with high weights. Algorithm 2 shows a pseudocode of the resampling algorithm of Channel-SLAM which is based on the systematic resampling algorithm [49]. Similarly to Algorithm 1, the Channel-SLAM resampling algorithm consists of a resampling algorithm for the superPF which includes resampling algorithms for the subPFs. First, the estimated sampled cumulative distribution function (CDF) of the superPF is constructed, presented by a vector $\mathbf{c}_p$ with length $N_p$ and element $[\mathbf{c}_p]_j$ with $j = 1, \ldots, N_p$. According to the estimated sampled CDF of the superPF, the subPFs are eliminated or resampled. The particles of the superPF with index $f$ are assigned to the resampled particle index $j$; see Algorithm 2, Line (10), for the assignment of the receiver state. Afterwards, the $(f, i)$th subPF is resampled with $i = 0, \ldots, N(t_k) - 1$ using a systematic resampling algorithm, where $\mathbf{c}_b$ represents the estimated sampled CDF of the $(f, i)$th subPF.

As mentioned before, whenever a new MPC is tracked, many particles are initialized to cover all possible VT positions in each subPF. For example if the $i$th MPC has a delay of $\widehat{d}_i(t_0) = 30$ m, each $i$th subPF would use $N_{P,j,i}(t_0) = 2821$ particles according to (29) for $j = 1, \ldots, N_p$ with spacing $\Delta_d = 1$ m. However during the receiver movement many particles of the subPFs are resampled at the same grid point because the states of the VTs $\mathbf{x}_{\mathrm{VT},i}(t_k)$ are time-invariant. In order to adapt the number of particles, we limit the number of resampled particles per grid point to $N_m$; see Algorithm 2, Line (20). Evaluations in Section 5 show that the reduction of the number of particles does not influence the positioning accuracy but, however, leads to a gain on computational performance. Afterwards, the states of the VTs $\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k)$ are drawn using a Gaussian-Kernel (cf. Line (26) in Algorithm 2). As stated in [31], the Gaussian-Kernel has a low bandwidth which does not influence the grid based reduction method.

Usually, we are interested in a point estimate of the state instead of its a posteriori PDF. According to [31], the MMSE point estimate, $\hat{\mathbf{x}}(t_k) = [\hat{\mathbf{x}}_u(t_k)^T, \hat{\mathbf{x}}_{\mathrm{VT}}(t_k)^T]^T$ (cf. Line (17) in Algorithm 1), is calculated by

$$\hat{\mathbf{x}}(t_k) \approx \sum_{j=1}^{N_s} w^{(j)}(t_k)$$

**Input**:

States and weights:

$$\left\{\mathbf{x}_u^{(j)}(t_k), w^{(j)}(t_k), \left\{\left\{\mathbf{x}_{\mathrm{VT},i}^{(j,a)}(t_k), w_i^{(j,a)}(t_k)\right\}_{a=1}^{N_{p,j,i}(t_k)}\right\}_{i=0}^{N(t_k)-1}\right\}_{j=1}^{N_s}$$

**Output:**

Resampled states and weights:

$$\left\{\widetilde{\mathbf{x}}_u^{(j)}(t_k), \widetilde{w}^{(j)}(t_k), \left\{\left\{\widetilde{\mathbf{x}}_{\mathrm{VT},i}^{(j,a)}(t_k), \widetilde{w}_i^{(j,a)}(t_k)\right\}_{a=1}^{N_{p,j,i}(t_k)}\right\}_{i=0}^{N(t_k)-1}\right\}_{j=1}^{N_s}$$

(1) Initialize the CDF for superPF: $[\mathbf{c}_p]_1 = w^{(1)}(t_k)$;
(2) **for** $j = 2 : N_s$ **do**
(3) Construct CDF for superPF:
$[\mathbf{c}_p]_j = [\mathbf{c}_p]_{j-1} + w^{(j)}(t_k)$;
(4) $f = 1$;
(5) Draw starting point: $[\mathbf{u}_p]_1 \sim \mathcal{U}[0, N_s^{-1}]$;
(6) **for** $j = 1 : N_s$ **do**
(7) $[\mathbf{u}_p]_j = [\mathbf{u}_p]_1 + N_s^{-1}(j-1)$;
(8) **while** $[\mathbf{u}_p]_j > [\mathbf{c}_p]_f$ **do**
(9) $f = f + 1$;
(10) Assign: $\{\widetilde{\mathbf{x}}_u^{(j)}(t_k), \widetilde{w}^{(j)}(t_k)\} = \{\mathbf{x}_u^{(f)}(t_k), 1/N_s\}$;
(11) **for** $i = 0 : N(t_k) - 1$ **do**
(12) Initialize the CDF for $(f, i)$-th subPF:
$[\mathbf{c}_b]_1 = w_i^{(f,1)}(t_k)$;
(13) **for** $a = 2 : N_{P,i,f}(t_k)$ **do**
(14) Construct CDF for subPF:
$[\mathbf{c}_b]_a = [\mathbf{c}_b]_{a-1} + w_i^{(f,a)}(t_k)$;
(15) $g = 1, \quad a_r = 1$;
(16) **for** $a = 1 : N_{P,f,i}(t_k)$ **do**
(17) $[\mathbf{u}_b]_a = [\mathbf{u}_b]_1 + \dfrac{1}{N_{P,i,f}(t_k)\ (a-1)}$;
(18) **while** $[\mathbf{u}_b]_a > [\mathbf{c}_b]_g$ **do**
(19) $g = g + 1$;
(20) **if** $B(\widetilde{\mathbf{x}}_{\mathrm{VT},i}^{(j,a_r)}(t_k)) \le N_m$ **then**
(21) Assign: $\widetilde{\mathbf{x}}_{\mathrm{VT},i}^{(j,a_r)}(t_k) = \mathbf{x}_{\mathrm{VT},i}^{(f,g)}(t_k)$;
(22) $a_r = a_r + 1$;
(23) $B\left(\widetilde{\mathbf{x}}_{\mathrm{VT},i}^{(j,a_r)}(t_k)\right) = B\left(\widetilde{\mathbf{x}}_{\mathrm{VT},i}^{(j,a_r)}(t_k)\right) + 1$;
(24) Update number of particles: $N_{P,j,i}(t_k) = a_r$;
(25) **for** $a = 1 : N_{P,j,i}(t_k)$ **do**
(26) Draw $\widetilde{\mathbf{x}}_{\mathrm{VT},i}^{(j,a)}(t_k)$ from the Gaussian-Kernel;
(27) Assign: $\widetilde{w}_i^{(j,a)}(t_k) = 1/N_{P,j,i}(t_k)$;

Algorithm 2: Channel-SLAM resampling algorithm.

$$\times \begin{bmatrix} \mathbf{x}_u^{(j)}(t_k) \\ \sum_{a=1}^{N_{P,j,0}(t_k)} w_0^{(j,a)}(t_k)\, \mathbf{x}_{\mathrm{VT},0}^{(j,a)}(t_k) \\ \vdots \\ \sum_{a=1}^{N_{P,j,N(t_k)-1}(t_k)} w_{N(t_k)-1}^{(j,a)}(t_k)\, \mathbf{x}_{\mathrm{VT},N(t_k)-1}^{(j,a)}(t_k) \end{bmatrix}. \tag{31}$$

## 4. Posterior Cramér-Rao Lower Bound

The PCRLB can be calculated by the inverse of the posterior information matrix $\mathbf{J}(t_k)$ and provides a lower bound of the variance of a Bayesian estimator [50] with

$$\begin{aligned} &\mathbf{E}\left[\left(\widehat{\mathbf{x}}(t_k) - \mathbf{x}(t_k)\right)\left(\widehat{\mathbf{x}}(t_k) - \mathbf{x}(t_k)\right)^T\right] = \mathbf{M}(t_k) \\ &\quad \ge \mathbf{J}(t_k)^{-1}. \end{aligned} \tag{32}$$

The inequality in (32) means that the difference $\mathbf{M}(t_k) - \mathbf{J}(t_k)^{-1}$ is a positive semidefinite matrix. For the performance evaluation of a filter like Channel-SLAM with the system equations of (10) and (11), the posterior information matrix can be calculated recursively according to [51], with

$$\begin{aligned}\mathbf{J}(t_k) &= \mathbf{D}_{22}(t_k) \\ &\quad - \mathbf{D}_{21}(t_k)\left(\mathbf{J}(t_{k-1}) + \mathbf{D}_{11}(t_k)\right)^{-1}\mathbf{D}_{12}(t_k),\end{aligned} \tag{33}$$

where

$$\begin{aligned}\mathbf{D}_{11}(t_k) &= -\mathbf{E}\left[\Delta^{\mathbf{x}(t_{k-1})}_{\mathbf{x}(t_{k-1})} \ln \mathrm{p}\left(\mathbf{x}(t_k) \mid \mathbf{x}(t_{k-1}), \dot{\Psi}(t_k)\right)\right], \\ \mathbf{D}_{21}(t_k) &= -\mathbf{E}\left[\Delta^{\mathbf{x}(t_{k-1})}_{\mathbf{x}(t_k)} \ln \mathrm{p}\left(\mathbf{x}(t_k) \mid \mathbf{x}(t_{k-1}), \dot{\Psi}(t_k)\right)\right] \\ &= \mathbf{D}_{12}(t_k)^T, \\ \mathbf{D}_{22}(t_k) &= -\mathbf{E}\left[\Delta^{\mathbf{x}(t_k)}_{\mathbf{x}(t_k)} \ln \mathrm{p}\left(\mathbf{x}(t_k) \mid \mathbf{x}(t_{k-1}), \dot{\Psi}(t_k)\right)\right] \\ &\quad - \mathbf{E}\left[\Delta^{\mathbf{x}(t_k)}_{\mathbf{x}(t_k)} \ln \mathrm{p}\left(\mathbf{z}(t_k) \mid \mathbf{x}(t_k)\right)\right],\end{aligned} \tag{34}$$

where $\nabla_a$ stands for the first-order partial derivatives with respect to $a$ and $\Delta^b_a$ stands for the second-order partial derivatives with $\Delta^b_a \triangleq \nabla_a \nabla_b^T$. In order to calculate the PCRLB, we use the transition model of Section 3.2.1. In case of non-Gaussian noise and nonlinearity as in Section 3.2.2, the expectation estimator in (34) has to be approximated by Monte Carlo simulations. To calculate the PCRLB, we combine the time-invariant transition model for the VTs $\mathbf{x}_{\mathrm{VT}}(t_k)$ as introduced in (14) and the transition model of the receiver (35), with

$$\mathbf{x}(t_k) = \underbrace{\begin{pmatrix}\mathbf{A}_u\left(t_\delta, \dot{\Psi}(t_k)\right) & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\end{pmatrix}}_{\mathbf{A}(t_\delta, \dot{\Psi}(t_k))}\mathbf{x}(t_{k-1}) + \mathbf{n}_t(t_k), \tag{35}$$

where $\mathbf{n}_t(t_k) \sim \mathcal{N}(0, \mathbf{Q}(t_\delta))$ is the transition noise with covariance matrix

$$\mathbf{Q}(t_\delta) = \begin{pmatrix}\mathbf{Q}_u(t_\delta) & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{pmatrix}, \tag{36}$$

where $\mathbf{Q}_u(t_\delta)$ is defined in (21).

Under the condition of a known control input $\dot{\Psi}(t_k)$, see [51], and using the linear transition model of Section 3.2.1 and Gaussian distributed transition noise of (36), we obtain for (34)

$$\begin{aligned}\mathbf{D}_{11}(t_k) &= \mathbf{A}\left(t_\delta, \dot{\Psi}(t_k)\right)^T \mathbf{Q}(t_\delta)^{-1}\mathbf{A}\left(t_\delta, \dot{\Psi}(t_k)\right), \\ \mathbf{D}_{12}(t_k) &= -\mathbf{A}\left(t_\delta, \dot{\Psi}(t_k)\right)^T \mathbf{Q}(t_\delta)^{-1} \\ \mathbf{D}_{22}(t_k) &= \mathbf{Q}(t_\delta)^{-1} + \mathbf{E}\left[\mathbf{F}(t_k)\right],\end{aligned} \tag{37}$$

where the matrix $\mathbf{F}(t_k)$ is the snapshot based Fisher information matrix. Substituting (37) into (33), we obtain

$$\begin{aligned}\mathbf{J}(t_k) &= \mathbf{E}\left[\mathbf{F}(t_k)\right] + \Big(\mathbf{Q}(t_\delta) \\ &\quad + \mathbf{A}\left(t_\delta, \dot{\Psi}(t_k)\right)\mathbf{J}(t_{k-1})^{-1}\mathbf{A}\left(t_\delta, \dot{\Psi}(t_k)\right)^T\Big)^{-1},\end{aligned} \tag{38}$$

using the matrix inversion lemma because of the singularity of $\mathbf{Q}(t_\delta)$.

The snapshot based Fisher information matrix $\mathbf{F}(t_k)$ in (38) can be obtained by

$$\left[\mathbf{F}(t_k)\right]_{k,w} = \left\{\frac{\partial \boldsymbol{\mu}(\mathbf{x}(t_k))^H}{\partial [\mathbf{x}(t_k)]_k}\mathbf{R}^{-1}(t_k)\frac{\partial \boldsymbol{\mu}(\mathbf{x}(t_k))}{\partial [\mathbf{x}(t_k)]_w}\right\}, \tag{39}$$

with the covariance matrix $\mathbf{R}(t_k)$, and

$$\mu(\mathbf{x}(t_k)) = \left[d_0(t_k), \ldots, d_{N(t_k)-1}(t_k)\right], \tag{40}$$

with the propagation lengths $d_i(t_k)$ of (16) for the $i$th MPC.

## 5. Evaluations Based on Measurements

This section evaluates the derived algorithm based on channel measurements on an airfield in front of a hangar with a single static physical transmitter and a moving pedestrian as shown in Figure 6. Figure 6 provides the scenario by a top view with the physical transmitter position in red, the track in blue, the starting position in green, and the end position in magenta. The measurements are conducted using the MEDAV RUSK-DLR broadband channel sounder in single-input single-output (SISO) mode with the measurement setup as indicated in Figure 7. The transmitter and receiver are connected to the same rubidium clock to prevent time drifts during the measurements. The static physical transmitter emits a 10 mW multitone signal (see [52]) with $N = 1281$ subcarriers having equal gains at a center frequency of 1.51 GHz and a bandwidth of $B = 100$ MHz. On the receiver side, the CIR snapshots are repeatedly measured every $T_g = 1.024$ ms. As shown in Figure 7, the receiving antenna is mounted on a pole attached on the backpack of the pedestrian. Additionally, the pedestrian is equipped with a hand-held equipment including a Xsense IMU (MTI-G-700) and a laptop which stores the IMU measurements. In order to obtain the ground truth of the pedestrians movement, a prism is mounted next to the antenna at the pole above the pedestrian. The prism is tracked by a tachymeter (TPS1200 from Leica Geosystems AG) which sends the measured coordinates to the laptop that records the coordinates simultaneously with the IMU measurements. The tachymeter gives a nominal accuracy in the subcentimeter domain. To synchronize all devices, the laptop is additionally connected by cable to the channel sounder. Thus, we are able to obtain the ground truth of the pedestrian for each captured CIR snapshot. Although the synchronization between the IMU and the channel sounder might be in the ms scale only, the influence on the position estimation is negligible because of the low pedestrian speed of around 0.7 m/s.

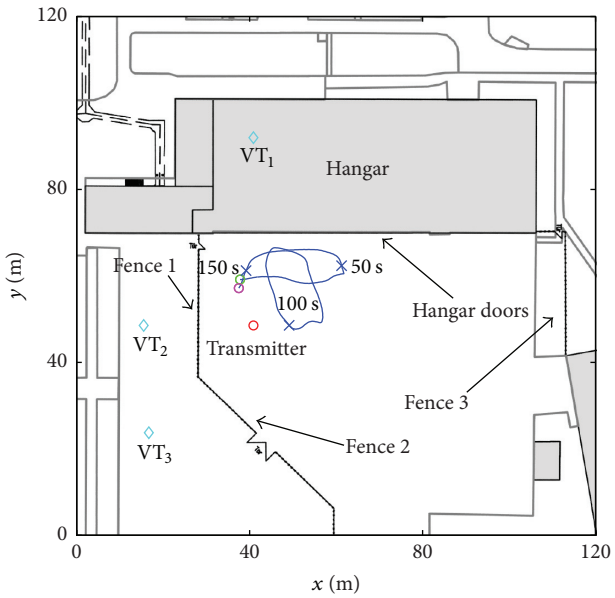



Figure 6: Measurement scenario with a fixed transmitter and a moving receiver (pedestrian). The pedestrian moves on the blue track for 155 s, in total 111 m. The starting position and end position are indicated by the green and magenta circles. The metalized doors of the hangar and the chain-link fences act as reflecting surfaces for the transmitted wireless signal. Hence, we can calculate the corresponding VT positions by mirroring the physical transmitter position on the reflecting surfaces.

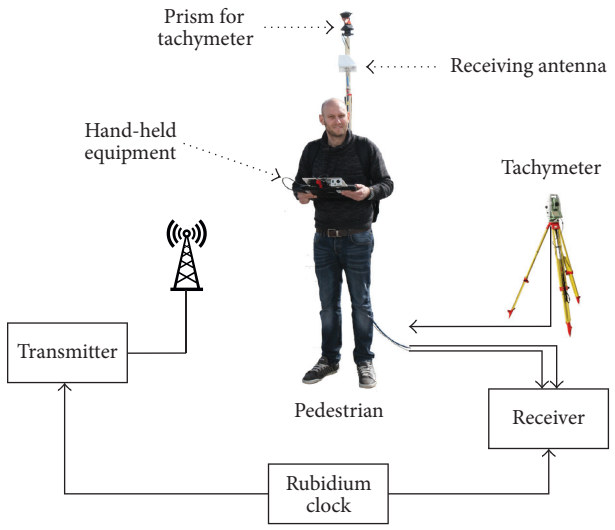



Figure 7: Measurement setup: transmitter and receiver use the same rubidium clock for synchronization. The receiving antenna is mounted on a stick next to a prism for measuring the ground truth of the moving pedestrian. Additionally, the pedestrian is holding a hand-held device which consists of an IMU and a laptop.

The pedestrian is moving on the indicated blue track of Figure 6 for 155 s or 111 m in front of a hangar with metalized doors. During the whole pedestrian movement, the LoS path between transmitter and receiver is present. Figure 8 shows the recorded CIRs versus the pedestrian moving time in seconds, where the time delays of the CIRs are multiplied by the speed of light, thus, in meters.

In order to exploit the multipath propagation for positioning, we have to estimate and track the MPCs over time. Hence, the accuracy of Channel-SLAM relies directly on the accuracy of the CIR estimations of KEST. Channel-SLAM considers an underdetermined system; therefore, long visible paths are preferable. Thus for the evaluations, we extract from the KEST estimates only the long visible paths as visualized in Figure 9 (Channel-SLAM could use all detected MPCs; however, this would increase the computational complexity). Figure 9 shows the estimation results of KEST for the CIR versus the pedestrian moving time in seconds. The black circled line in Figure 9 indicates the geometrical line-of-sight (GLoS) path delay, which matches perfectly with the

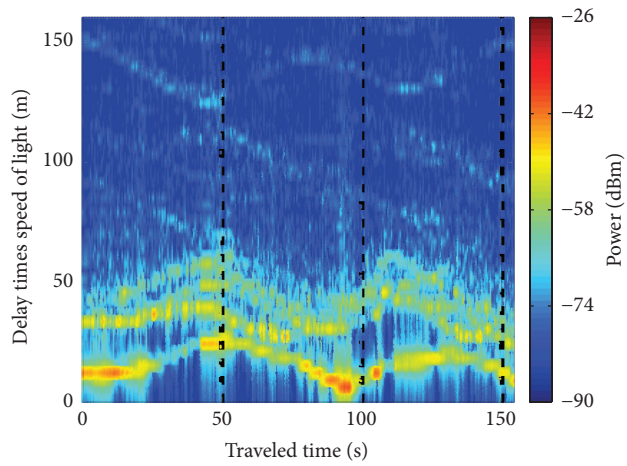



Figure 8: Recorded CIRs versus the pedestrian moving time in seconds.

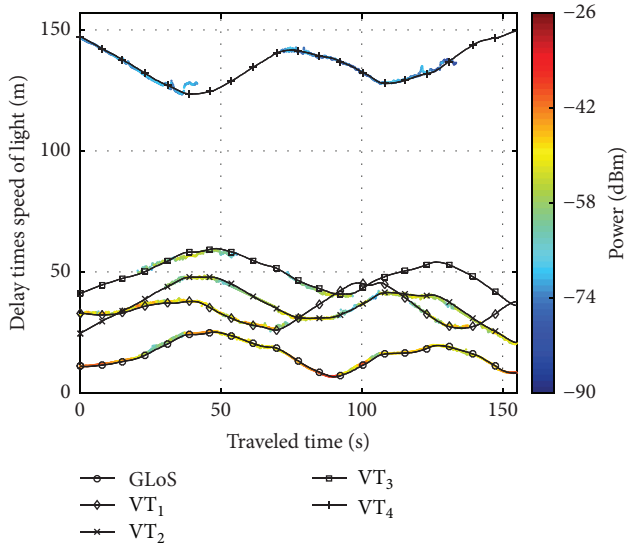



Figure 9: Estimation results of KEST for the CIR versus the pedestrian moving time in seconds where only long tracked paths are visualized. Additionally it shows the calculated propagation paths of the GLoS path and paths of $VT_1$, $VT_2$, $VT_3$, and $VT_4$.

KEST estimates for the first path. Additionally, other MPCs can be tracked by KEST for a long time. The metalized doors of the hangar act as a reflecting surface for the transmitted wireless signal. We can obtain the position of $VT_1$ by mirroring the physical transmitter on the reflecting surface as mentioned in Section 2. Additionally, the chain-link fences indicated by Fence 1, Fence 2, and Fence 3 act as reflecting surfaces. We obtain $VT_2$, $VT_3$, and $VT_4$ by mirroring the physical transmitter position at Fence 1, Fence 2, and Fence 3. The positions of the hangar, Fence 1, Fence 2, and Fence 3 are measured using the tachymeter; thus, we are able to calculate the positions of $VT_1$, $VT_2$, $VT_3$, and $VT_4$. Please note that $VT_4$ is not shown in Figure 6. Based on the calculated VT positions, we are able to calculate the hypothetical propagation distances between these VTs and the moving pedestrian. We can see that they match the KEST estimates as indicated by the black lines in Figure 9. The measurement scenario considers only one time signal reflections. For examples on VTs with multiple number of reflections, diffractions, or scattering, please see [31].

The evaluations are performed using $N_s = 2000$ particles in the superPF, whereas the number of particles for the subPFs for each MPC is different depending on the estimated delay of each MPC. Channel-SLAM obtains the measurements $\mathbf{z}(t_k)$ from KEST and the heading change $\dot{\Psi}(t_k)$ from the IMU every $t_\delta = 0.1$ s. For the initialization of Channel-SLAM, we use prior information $\mathrm{p}(\mathbf{x}_u(t_0))$. The prior information includes the starting position and moving direction, whereas the speed is initialized using a uniform distribution between 0 m/s and 1 m/s. Please note that an unknown starting position and direction or larger initial uncertainties may result in a biased and rotated coordinate system in the estimation. We empirically set $\Delta_d = 1$ m. Since Channel-SLAM has no knowledge of the physical transmitter position, Channel-SLAM estimates the position of the physical transmitter as a VT. During the pedestrian

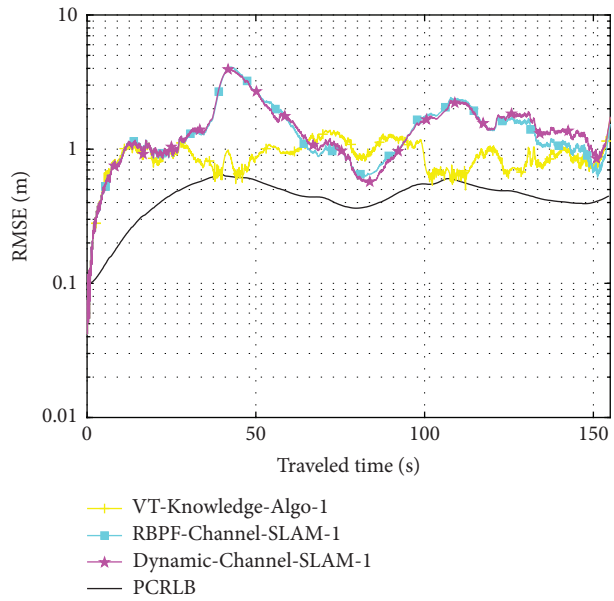



FIGURE 10: $\mathrm{RMSE}_u(t_k)$ versus the pedestrian moving time in seconds for Dynamic-Channel-SLAM-1, RBPF-Channel-SLAM-1, VT-Knowledge-Algo-1, and the PCRLB.

movement, the number of tracked MPCs changes which results in removing and initialization of subPFs during the movement. Additionally, because Channel-SLAM does not consider retracking of previous MPCs, for example, the MPCs of $\mathrm{VT}_1$ and $\mathrm{VT}_3$ which are tracked multiple times, they are initialized without any prior information.

To see the positioning performance of Channel-SLAM, we compare the following algorithms and bounds:

(i) Dynamic-Channel-SLAM: it is a Channel-SLAM implementation using the dynamical adaptation of the number of particles as introduced in Section 3.4 where we limit the number of particles per bin to $N_m = 30$ and the grid size to $\Delta_d = 1$ m.

(ii) RBPF-Channel-SLAM: it is similar to Dynamic-Channel-SLAM, however, without using the dynamical adaptation of the number of particles.

(iii) VT-Knowledge-Algo: it is a positioning algorithm with perfect knowledge of all VT positions $\mathbf{r}_{\mathrm{VT},i}(t_k)$ and additional propagation lengths $d_{\mathrm{VT},i}$. Because the measurement scenario considers only one time reflections, VT-Knowledge-Algo reflects algorithms of [24, 53] which consider reflected signals as signals emitted from VTs, where the VT positions are precalculated based on the knowledge of the reflecting surface and physical transmitter positions. VT-Knowledge-Algo can be seen as a lower bound for Channel-SLAM. Similarly to Channel-SLAM, VT-Knowledge-Algo uses the delays of the estimated MPCs provided by KEST as input, assumes the knowledge of starting position and direction, and is implemented using a PF with $N_s = 2000$ particles.

(iv) PCRLB: it is as the PCRLB derived in Section 4 using $\sigma^2_{q_u} = 5 \cdot 10^{-4}\ \mathrm{m}^2/\mathrm{s}^3$, the standard deviation $\boldsymbol{\sigma}_z(t_k)$ for each MPC from KEST, and the prior like the Channel-SLAM algorithms.

The above-mentioned algorithms are evaluated using the Gaussian-Transition-Model of Section 3.2.1 indicated by index 1 and the Rician-Transition-Model of Section 3.2.2 indicated by index 2, for example, RBPF-Channel-SLAM-1 and RBPF-Channel-SLAM-2 for RBPF-Channel-SLAM. For the Gaussian-Transition-Model we set the continuous-time process noise intensity to $\sigma^2_{q_u} = 5 \cdot 10^{-4}\ \mathrm{m}^2/\mathrm{s}^3$ and for the Rician-Transition-Model we set the standard deviation of the heading noise of (25) to $\sigma_\Phi(t_k) = 0.1^\circ$ and the scale parameter of the velocity of (24) to $\sigma_m(t_k) = 0.025$ m/s.

Figure 10 shows the root mean square error (RMSE) $\mathrm{RMSE}_u(t_k) = \sqrt{\mathbf{E}\{\|\mathbf{r}_u(t_k) - \hat{\mathbf{r}}_u(t_k)\|^2\}}$ of the estimated pedestrian position $\hat{\mathbf{r}}_u(t_k)$ versus the pedestrian moving time for Dynamic-Channel-SLAM-1 in magenta, RBPF-Channel-SLAM-1 in cyan, and VT-Knowledge-Algo-1 in yellow and the black line indicates the PCRLB. Because the PF includes randomness, the position estimates differ for each evaluation unless the number of particles is infinite even if the same measurement data are used. Therefore, we perform 200 independent evaluations based on the same measurement data. For the evaluations we add an artificial clock bias to the measurements to verify the clock bias estimation capabilities. Because of the initialization of the receiver position using prior knowledge, all algorithms perform similarly at the beginning of the track where the position error is rather low. Afterwards, $\mathrm{RMSE}_u(t_k)$ is varying between 0.6 m and 4 m for Dynamic-Channel-SLAM-1 and RBPF-Channel-SLAM-1. VT-Knowledge-Algo-1 can be interpreted as a lower bound and estimates the receiver position with the lowest RMSE. Between 70 s and 90 s of the receiver movement, VT-Knowledge-Algo-1 has a slightly higher RMSE which might

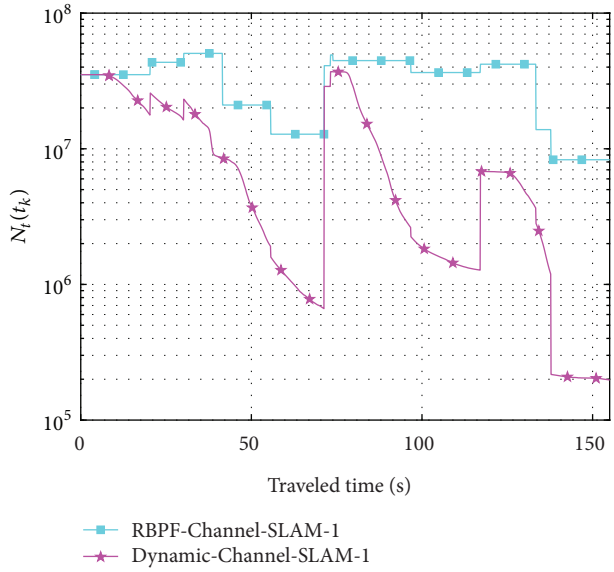



FIGURE 11: Total number of particles $N_t(t_k)$ versus the pedestrian moving time in seconds for Dynamic-Channel-SLAM-1 and RBPF-Channel-SLAM-1.

be due to the nonperfect reflecting surfaces, KEST estimation errors, or small inaccuracies in the calculations of the VT positions. Furthermore, we see that we obtain a similar RMSE for Dynamic-Channel-SLAM-1 and RBPF-Channel-SLAM-1. However, if we have a look on the number of used particles, as shown in Figure 11, we see a major computational performance gain of the Dynamic-Channel-SLAM-1 compared to RBPF-Channel-SLAM-1. Figure 11 shows the total number of particles $N_t(t_k)$ calculated according to (30) for $N_p = 2000$ versus the pedestrian moving time. At the beginning, both Channel-SLAM algorithms are initialized with the same number of particles. As soon as the pedestrian is moving, the estimations of the VT positions are converging resulting in a reduction of the number of particles for Dynamic-Channel-SLAM-1. When new MPCs are tracked (see Figures 9 and 11), the total number of particles $N_t(t_k)$ increases and reduces afterwards during runtime. Especially at the end of the track, Dynamic-Channel-SLAM-1 uses 40 times less particles than RBPF-Channel-SLAM-1.

As mentioned before, the black line in Figure 10 indicates the PCRLB. The PCRLB shows the theoretical performance bound which has on average a 2-3 times lower RMSE than Dynamic-Channel-SLAM-1 and RBPF-Channel-SLAM-1. However, the curve shapes of the PCRLB, Dynamic-Channel-SLAM-1, and RBPF-Channel-SLAM-1 are similar. By increasing the number of particles, the RMSE of Dynamic-Channel-SLAM-1 and RBPF-Channel-SLAM-1 might be decreased. Additionally, the PCRLB shows a theoretical limit which is not affected by estimation inaccuracies of KEST caused by nonperfect reflecting surfaces or DMCs.

Figure 12 shows the RMSE for Dynamic-Channel-SLAM-1 in magenta, RBPF-Channel-SLAM-1 in cyan, VT-Knowledge-Algo-1 in yellow, Dynamic-Channel-SLAM-2 in blue, RBPF-Channel-SLAM-2 in red, and VT-Knowledge-Algo-2 in green. Similarly to Figure 12, we see that we obtain a similar RMSE for Dynamic-Channel-SLAM-1 and RBPF-Channel-SLAM-1. Additionally, we see a significant performance gain in the position accuracy by comparing the different transition models. Similarly to Figure 12, VT-Knowledge-Algo-1 has between 80 s and 90 s a slightly higher RMSE because of the same reasons stated before. Furthermore, we see as well that we obtain a similar RMSE for Dynamic-Channel-SLAM-1 and RBPF-Channel-SLAM-1.

Figure 13 shows the enlarged measurement scenario with estimated PDFs for the physical transmitter, $VT_1$, $VT_2$, and the pedestrian position. Whereas the PDFs of the physical transmitter, $VT_2$, and pedestrian position are the estimation results at the end of the track, the PDFs of $VT_1$ are the estimation results when the tracking of the MPC is lost, that is, after 75 s. We see that especially the physical transmitter and $VT_2$ position can be estimated with a low uncertainty. Additionally, Figure 13 shows two examples of the MMSE point estimates of the receiver position for Dynamic-Channel-SLAM-2 in red, VT-Knowledge-Algo-2 in green, and ground truth of the track in blue. We see that we obtain similar position estimation results for both algorithms.

## 6. Conclusions

In this paper, we extended the work on multipath assisted positioning, called Channel-SLAM. The new positioning method takes advantage of the multipath components (MPCs) instead of mitigating them. In the proposed approach, multipath signals are treated as signals from virtual transmitters (VTs), where the locations of these VTs are unknown. To improve the accuracy of Channel-SLAM, an inertial measurement unit (IMU) is used to obtain heading information of the moving receiver. Furthermore, we investigate a novel particle filter (PF) implementation which adapts the number of particles during runtime. Measurement

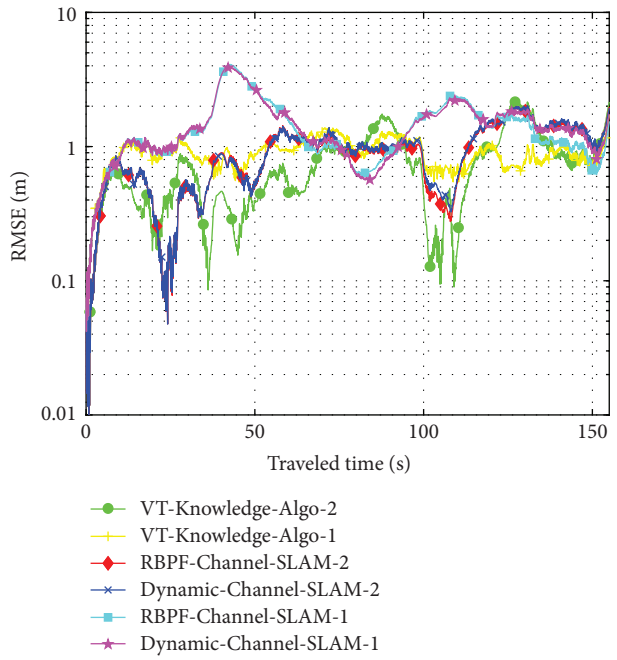



FIGURE 12: $\mathrm{RMSE}_u(t_k)$ versus the pedestrian moving time in seconds for Dynamic-Channel-SLAM-1, RBPF-Channel-SLAM-1, VT-Knowledge-Algo-1, Dynamic-Channel-SLAM-2, RBPF-Channel-SLAM-2, and VT-Knowledge-Algo-2.

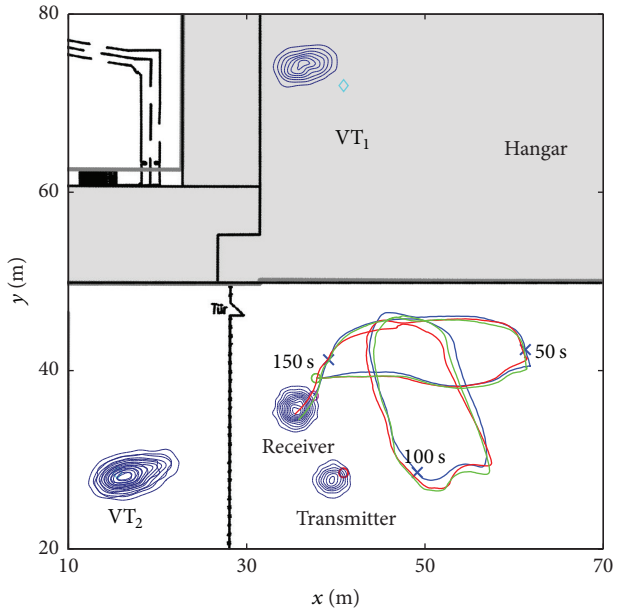



FIGURE 13: Enlarged measurement scenario of Figure 6 with the ground truth of the track in blue. The figure shows two examples of the MMSE point estimates of the receiver position for Dynamic-Channel-SLAM-2 in red and VT-Knowledge-Algo-2 in green. The starting position and end position are indicated by the green and magenta circles. Additionally, it shows the estimated PDFs for the physical transmitter, $\mathrm{VT}_1$, $\mathrm{VT}_2$, and pedestrian position.

results show that accurate position estimation is possible without the knowledge of the physical transmitter position. Hence, the new algorithm does not rely on prior information such as the room layout or information collected in a database for fingerprinting except for the initial position and direction. We compare the position accuracy of Channel-SLAM to that of a derived posterior Cramér-Rao lower bound (PCRLB). Additionally, we obtain similar position estimation results with Channel-SLAM and an algorithm with perfect knowledge of all VT positions.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This work has been performed in the framework of the DLR project *Navigation 4.0* and the European Union Horizon 2020 Research and Innovation Programme under Grant Agreement no. 636537, *HIGHTS (*High Precision Positioning for Cooperative-ITS Applications*)*.

## References

[1] S. Banville and F. Van Diggelen, *Innovation: Precise Positioning Using Raw Gps Measurements from Android Smartphones*, GPS World, 2016.

[2] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 37, no. 6, pp. 1067–1080, 2007.

[3] L. Mainetti, L. Patrono, and I. Sergi, "A survey on indoor positioning systems," in *Proceedings of the 22nd International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2014*, pp. 111–120, September 2014.

[4] D. Dardari, P. Closas, and P. M. Djuric, "Indoor tracking: theory, methods, and technologies," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1261-1262, 2015.

[5] N. Li and B. Becerik-Gerber, "Performance-based evaluation of RFID-based indoor location sensing solutions for the built environment," *Advanced Engineering Informatics*, vol. 25, no. 3, pp. 535–546, 2011.

[6] A. H. Sayed, A. Tarighat, and N. Khajehnouri, "Network-based wireless location: challenges faced in developing techniques for accurate wireless location information," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 24–40, 2005.

[7] Y. Zhao, "Standardization of mobile phone positioning for 3G systems," *IEEE Communications Magazine*, vol. 40, no. 7, pp. 108–116, 2002.

[8] K. Kaemarungsi and P. Krishnamurthy, "Properties of indoor received signal strength for WLAN location fingerprinting," in *Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS '04)*, pp. 14–23, August 2004.

[9] M. Z. Win and R. A. Scholtz, "Characterization of ultra-wide bandwidth wireless indoor channels: A communication-theoretic view," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 9, pp. 1613–1627, 2002.

[10] A. F. Molisch, D. Cassioli, and C.-C. Chong, "A comprehensive standardized model for ultrawideband propagation channels," *IEEE Transactions on Antennas and Propagation*, vol. 54, no. 11, pp. 3151–3166, 2006.

[11] C. Steiner and A. Wittneben, "Low complexity location fingerprinting with generalized UWB energy detection receivers," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, part 2, pp. 1756–1767, 2010.

[12] B. W. Parkinson and J. J. Spilker Jr., *Global Positioning System:Theory and Applications*, American Institute of Aeronautics and Astronautics Inc., 1996.

[13] A. J. van Dierendonck, P. Fenton, and T. Ford, "Performance Evaluation of the Multipath Estimating Delay Lock Loop," *Journal of the Institute of Navigation*, vol. 39, no. 3, pp. 265–284, 1992.

[14] L. Garin, F. van Diggelen, and J. M. Rousseau, *Strobe & Edge Correlator Multipath Mitigation for Code*, 1996.

[15] G. A. McGraw and M. S. Braasch, *GNSS Multipath Mitigation using Gated and High Resolution Correlator Concepts*, 1999.

[16] B. R. Townsend, P. C. Fenton, K. J. Van Dierendonck, and D. J. R. Van Nee, "Performance Evaluation of the Multipath Estimating Delay Lock Loop," *Navigation*, vol. 42, no. 3, pp. 502–514, 1995.

[17] F. Antreich, J. A. Nossek, and W. Utschick, "Maximum likelihood delay estimation in a navigation receiver for aeronautical applications," *Aerospace Science and Technology*, vol. 12, no. 3, pp. 256–267, 2008.

[18] B. Krach, P. Robertson, and R. Weigel, "An efficient two-fold marginalized bayesian filter for multipath estimation in satellite navigation receivers," *Eurasip Journal on Advances in Signal Processing*, vol. 2010, Article ID 287215, 2010.

[19] P. Closas, C. Fernández-Prades, and J. A. Fernández-Rubio, "A bayesian approach to multipath mitigation in GNSS receivers," *IEEE Journal on Selected Topics in Signal Processing*, vol. 3, no. 4, pp. 695–706, 2009.

[20] W. Wang, T. Jost, C. Gentner, S. Zhang, and A. Dammann, "A semiblind tracking algorithm for joint communication and ranging with OFDM signals," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 7, pp. 5237–5250, 2016.

[21] M. Triki, D. T. M. Slock, V. Rigal, and P. François, "Mobile terminal positioning via power delay profile fingerprinting: Reproducible validation simulations," in *Proceedings of the 2006 IEEE 64th Vehicular Technology Conference, VTC-2006 Fall*, pp. 2840–2844, September 2006.

[22] E. Kupershtein, M. Wax, and I. Cohen, "Single-site emitter localization via multipath fingerprinting," *IEEE Transactions on Signal Processing*, vol. 61, no. 1, pp. 10–21, 2013.

[23] Y. Shen and M. Z. Win, "On the use of multipath geometry for wideband cooperative localization," in *Proceedings of the 2009 IEEE Global Telecommunications Conference, GLOBECOM 2009*, December 2009.

[24] P. Meissner, E. Leitinger, and K. Witrisal, "UWB for robust indoor tracking: weighting of multipath components for efficient estimation," *IEEE Wireless Communications Letters*, vol. 3, no. 5, pp. 501–504, 2014.

[25] M. Zhu, J. Vieira, Y. Kuang, K. Åström, A. F. Molisch, and F. Tufvesson, "Tracking and positioning using phase information from estimated multi-path components," in *Proceedings of the IEEE International Conference on Communication Workshop, ICCW 2015*, pp. 712–717, June 2015.

[26] Y. Kuang, K. Astrom, and F. Tufvesson, "Single antenna anchor-free UWB positioning based on multipath propagation," in *Proceedings of the IEEE International Conference on Communications (ICC '13)*, pp. 5814–5818, IEEE, Budapest, Hungary, June 2013.

[27] C. Gentner and T. Jost, "Indoor positioning using time difference of arrival between multipath components," in *Proceedings of the 2013 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2013*, Montbeliard, France, October 2013.

[28] C. Gentner, T. Jost, and A. Dammann, "Accurate Indoor Positioning using Multipath Components," in *Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation*, Nashville, TN, USA, September 2013.

[29] C. Gentner, R. Pöhlmann, T. Jost, and A. Dammann, "Multipath assisted positioning using a single antenna with angle of arrival estimations," in *Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation*, Tampa, FL, USA, September 2014.

[30] C. Gentner, R. Pöhlmann, M. Ulmschneider, T. Jost, and A. Dammann, "Multipath Assisted Positioning for Pedestrians," in *Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation*, Tampa, FL, USA, September 2015.

[31] C. Gentner, T. Jost, W. Wang, S. Zhang, A. Dammann, and U.-C. Fiebig, "Multipath assisted positioning with simultaneous localization and mapping," *IEEE Transactions on Wireless Communications*, vol. 15, no. 9, pp. 6104–6117, 2016.

[32] O. J. Woodman, "An Introduction to Inertial Navigation," Tech. Rep. UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, 2007.

[33] P. Bello, "Characterization of randomly time-variant linear channels," *IEEE Transactions on Communications*, vol. 11, no. 4, pp. 360–393, 1963.

[34] G. L. Turin, F. D. Clapp, T. L. Johnston, S. B. Fine, and D. Lavry, "A statistical model of urban multipath propagation," *IEEE Transactions on Vehicular Technology*, vol. 21, no. 1, pp. 1–9, 1972.

[35] T. Jost, W. Wang, U.-C. Fiebig, and F. Pérez-Fontán, "Detection and tracking of mobile propagation channel paths," *Institute of Electrical and Electronics Engineers. Transactions on Antennas and Propagation*, vol. 60, no. 10, pp. 4875–4883, 2012.

[36] W. Wang, *Channel Measurement and Modeling for Mobile Radio Based Positioning [Ph.D. thesis]*, 2015.

[37] T. Jost, *Satellite-to-Indoor Wave Propagation for Positioning Applications [Ph.D. thesis]*, 2013.

[38] T. S. Rappaport, *Wireless Communications-Principles and Practice*, Prentice Hall, 1996.

[39] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[40] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[41] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, pp. 4282–4286, 1995.

[42] L. Yang, W. Fang, J. Li, R. Huang, and W. Fan, "Cellular automata pedestrian movement model considering human behavior," *Chinese Science Bulletin*, vol. 48, no. 16, pp. 1695–1699, 2003.

[43] T. Robin, G. Antonini, M. Bierlaire, and J. Cruz, "Specification, estimation and validation of a pedestrian walking behavior model," *Transportation Research Part B: Methodological*, vol. 43, no. 1, pp. 36–56, 2009.

[44] M. Khider, *Multisensor-Based Positioning for Pedestrian Navigation [Dissertation, thesis]*, University of Ulm, 2013.

[45] C. Jekeli, *Inertial Navigation Systems with Geodetic Applications*, Walter de Gruyter, 2001.

[46] W. T. Fong, S. K. Ong, and A. Y. C. Nee, "Methods for in-field user calibration of an inertial measurement unit without external equipment," *Measurement Science and Technology*, vol. 19, no. 8, Article ID 085202, 2008.

[47] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*, John Wiley & Sons, 2004.

[48] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond theKalman Filter: Particle Filters for Tracking Applications*, Artech House, 2004.

[49] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[50] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior cramér-rao bounds for discrete-time nonlinear filtering," *IEEE Transactions on Signal Processing*, vol. 46, no. 5, pp. 1386–1396, 1998.

[51] B. Siebler and S. Sand, "Posterior Cramér-Rao bound and suboptimal filtering for I$\mu$GNSS based cooperative train localization," in *Proceedings of the IEEE/ION Position, Location and Navigation Symposium, PLANS 2016*, pp. 353–358, April 2016.

[52] R. S. Thomä, M. Landmann, A. Richter, and U. Trautwein, *Smart Antennas-State of the Art, ser. EURASIP Book Series on SP&C*, Hindawi Publishing Corporation, 2005.

[53] E. Leitinger, P. Meissner, M. Lafer, and K. Witrisal, "Simultaneous localization and mapping using multipath channel information," in *Proceedings of the IEEE International Conference on Communication Workshop, ICCW 2015*, pp. 754–760, June 2015.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 3876906, 21 pages
https://doi.org/10.1155/2017/3876906

*Research Article*

# Semantic Labeling of User Location Context Based on Phone Usage Features

**Helena Leppäkoski,[1] Alejandro Rivero-Rodriguez,[1] Sakari Rautalin,[1] David Muñoz Martínez,[2] Jani Käppi,[3] Simo Ali-Löytty,[1] and Robert Piché[1]**

[1]*Tampere University of Technology, Tampere, Finland*
[2]*GE Healthcare, Helsinki, Finland*
[3]*HD Automotive Positioning Solutions at HERE, Tampere, Finland*

Correspondence should be addressed to Helena Leppäkoski; helena.leppakoski@tut.fi

Received 16 February 2017; Revised 27 May 2017; Accepted 13 July 2017; Published 24 August 2017

Academic Editor: Antonio de la Oliva



In mobile phones, the awareness of the user's context allows services better tailored to the user's needs. We propose a machine learning based method for semantic labeling that utilizes phone usage features to detect the user's home, work, and other visited places. For place detection, we compare seven different classification methods. We organize the phone usage data based on periods of uninterrupted time that the user has been in a certain place. We consider three approaches to represent this data: *visits*, *places*, and *cumulative samples*. Our main contribution is semantic place labeling using a small set of privacy-preserving features and novel data representations suitable for resource constrained mobile devices. The contributions include (1) introduction of novel data representations including accumulation and averaging of the usage, (2) analysis of the effect of the data accumulation time on the accuracy of the place classification, (3) analysis of the confidence on the classification outcome, and (4) identification of the most relevant features obtained through feature selection methods. With a small set of privacy-preserving features and our data representations, we detect the user's home and work with probability of 90% or better, and in 3-class problem the overall classification accuracy was 89% or better.

## 1. Introduction

The use of smartphones has dramatically changed during the last decade. Whereas only 1% of worldwide population owned a smartphone in 2006 [1], now the number has reached 24% [2]. Mobile phones have become the most personal computing device. Users carry them continuously throughout the day and expect them to deliver meaningful services on the move. In order to provide a more personal and relevant user experience, mobile services can benefit from knowledge about the user's context. Context sensing can deliver new ways in how people interact with mobile devices by making the devices appear to be more human and personal. Intelligent devices can recognize the user, adapt to the user and the user's context, and learn to be proactive.

The most well-known context-aware applications are location-based services [3]. The location is usually represented by a set of coordinates defining a point or area on the Earth. This representation does not provide direct information about the meaning and relevance of a place to the user. Although in some locations it may be possible to use reverse geocoding to infer the type of the place, it is difficult to infer the meaning of the place for each user as the same place can have different meaning for different people. For example, a gas station might mean a frequent visited place, a work place, or just a nearby place during the daily commute. By leveraging the sensing capabilities of today's mobile phones, it is feasible to build a model that provides context related information about the user location.

This work aims to provide a reliable method to infer the meaning of the visited places of mobile phone users. We propose a machine learning based method for semantic labeling that utilizes phone usage features to detect the user's home, work, and other visited places. Our proposal provides better understanding of the user's location context and allows mobile phones to deliver more personalized and intelligent services and applications to users. For example, applications that are aware of the user's semantic location could allow the user to set reminders to phone to trigger when leaving home, arriving to work, or going to a frequently visited place, or to set automatic functions based on current place, for example, changing profiles or silencing phone.

In this work we develop a system to learn and label a user's places based on phone usage and analyze the effects of different choices of data representation. Our goal is an automatic method for detecting places of a user by applying a classification model learned from the data of the other users. This is similar to a use case where the earlier users of an application have contributed to the model by providing their data, and later, using the model, the application labels the data of the new users. Our contributions include (1) the introduction of novel data representations including accumulation and averaging of the usage data and performance results based on the proposed data representations, (2) analysis of the effect of the data accumulation time on the accuracy of the place classification, (3) analysis of the confidence on the classification outcome, and (4) identification of the most relevant features obtained through feature selection methods.

For training and model assessment we use two data sets. One of these is the Mobile Data Challenge (MDC) database [4, 5], where about 200 users used Nokia N95 devices normally for time spans between 3 and 18 months. The data includes logs of phone calls and SMS, calendar entries, multimedia displayed, GPS information when available, network information, and system information (e.g., battery status, device inactive time). The other data set is smaller: it covers a shorter time span (1–3 months) and includes labeled data of 16 users. This data includes information on similar phone usage and activity patterns as the MDC data, but there are differences in what is measured and how the observations are processed before storing them, which also makes the available features different. Using the aforementioned data, we use supervised learning methods to create a place detection algorithm that estimates the semantic label of the current place based on the phone's current usage features.

The rest of this article is organized as follows. In Section 2 we outline the background of our work, highlighting the current needs for place detection. In Section 3 we present the data and features used in this work. Section 4 describes the methods used in the analyses and comparisons in this work: the data preprocessing and the data representations, different classification methods, the cross-validation method used in the comparisons, feature selection methods, and finally methods for assessing the confidence in the classification result. Section 5 presents the results of the analyses and comparisons. In Section 6, we discuss the findings of this work and summarize its similarities and differences to the related work. Finally, in Section 7 we conclude the article.

## 2. Related Work

Research on context-aware systems began in earnest in the early 1990s [10]. Context can refer to any information that can be used to characterize the situation of an entity, where an entity can be a person, place, or physical or computational object [11]. To infer a user's context, we use sensor information. Following Baldauf et al. [11], the notion of a sensor is generalized to encompass any data source. We distinguish three types of sensors.

*Physical sensors* are devices that detect and respond to input from the physical environment and capture physical data.

*Virtual sensors* capture contextual information from applications and services. They can be based on local services (e.g., calendar) or external services (e.g., weather forecast).

*Logical sensors* provide contextual information by combining information from physical and virtual sensors.

Most existing context-aware systems consider physical sensors [12], including the sensors related to the user's position, such as GPS, accelerometer, gyroscope (allowing, e.g., activity recognition) [13, 14], or sensors that measure the properties of the user's environment, such as magnetic field, light, or properties of various radio signals [15, 16]. Regarding virtual sensors, one of the most used sensors is the user's language. For instance, Google provides developers with the user's language through function `getDisplayLanguage` in the Android Developers API [17]. Other context related information can be provided to mobile applications in similar fashion.

Researchers have pointed out that, in addition to sensors, the usage of mobile phones can provide meaningful information about the user's context [6–8, 18]. Do and Gatica-Perez [18] assert that the user's context can be inferred based on the usage of applications (e.g., calls, e-mail, and web browser). Rahmati et al. use the smart phone's context information including time, day, movement information from accelerometer, cell ID location, and GPS location together with usage context (the prior visited web site, phone call, and application) to predict the next usage of the phone [19].

In this work we continue that line of research by studying the association between the usage of mobile phones and the user's context. More concretely, we investigate the main challenges and possible solutions for place detection, a particular case of semantic labeling. Place detection provides important information to improve context-aware applications.

Aiming at improving current place labeling techniques, we apply different supervised learning methods on mobile phone usage log data to find models that, based on the mobile phone usage patterns, allow assigning semantic labels to the places the user visits. The preliminary results of our work have been presented in [20]. This paper enhances the contributions of [20] in the following aspects: (1) we introduce here third data representation and cumulative samples; (2) in the analysis, we use two data sets instead of only one; (3) we provide results on the effect of the accumulation time of the cumulative samples on the accuracy of the classifier models that we use to provide the place labels; (4) we use sequential feature selection to decrease the computational load and

to improve the accuracy in the prediction phase when the classifier models are used to predict the place label; (5) we study the assessment of the confidence of the classification results; (6) we enhance the set of classification methods we used for the analysis to include also support vector machines and logistic regression.

Other works have been carried out with similar goals to ours [6–9, 21], that is, semantic place prediction, and use data derived from the same database as data set #1 in our work. They differ from our work in the following aspects: the number of features we used for our classification method is only 14 at most, while the other works use more features; we use different classifiers; while the other papers classify all the 10 labels available in data set #1, we prioritize recognizing *Home* and *Work* and therefore combine all the less frequent labels to one label *Other*; and we present a comparison between three different data representation schemes: visits, places, and cumulative samples. We also show that the accuracy can be improved by selecting a subset of the most relevant features to be used in the classification model and we study the benefit of rejecting classification results that obtain low confidence ranking from the classifier. Since its publication, the MDC data set has been extensively used in research. In addition to the semantic place prediction, it has been used, for example, in research of mobility patterns of phone users [22–24] and in human mobility prediction (prediction of the next location) [25, 26].

The research presented in this paper was conducted as part of the related work for the creation of the Place Monitor API of the Lumia SensorCore SDK [27]. The SDK is a collection of APIs to provide meaningful activity and location data from sensors that run constantly in the background in a low power mode.

## 3. Description of Data Sets

We used two different data sets for learning and predicting semantic place labels. In this section we describe the data and identify the most relevant features for place detection.

*3.1. Data Set #1.* Data set #1 is obtained from the MDC database made available by Idiap Research Institute, Switzerland, and owned by Nokia [4, 5]. The data set contains Nokia N95 smart phones usage data, collected by nearly 200 users over time periods that for many users exceed one year [5]. The information about the usage of the phones was automatically collected and anonymized. After the data collection, a clustering algorithm was used to identify the most relevant places for each user, that is, places that the user visited often and spent lot of time. These places the users labeled manually [9]. As our main focus was in the detection of *Home* and *Work* where people usually stay longer times, we extracted for our tests the data that was collected during the visits where the user stayed at least 20 minutes in the same place. The time intervals of these visits are defined in a database table `visits_20min.csv`, which is included in the MDC database, and it defines the start and end times, user ID, and place ID for more than 55,000 visits (see Figure 1). The place labels for the place IDs are defined in a separate

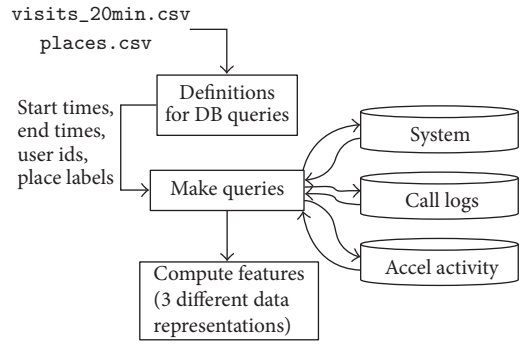



Figure 1: Obtaining data for feature computation.

MDC database table `places.csv`. The place in the database is defined so that it corresponds to a circle with 100 m radius [5].

Based on these data, we queried from the database the following phone usage data for each visit, that is, for a given user, all data entries between the start and end times of the visit:

*(i) System data*, including battery and charging status and counter for inactive time

*(ii) Call log*, including durations of each phone call

*(iii) Acceleration based activity data*, including accelerometer based estimates of the user's motion mode: *idle/still, walk, car/bus/motorbike, train/metro/tram, run, bicycle*, or *skateboard*. Due to the large area covered by a place, it is possible that the data from one place contains also significant amount of mobility, for example, walking or even being in a moving vehicle.

From these data entries, we computed for each visit the features to be used in the classification task. We decided to use only such sensor data that can be assumed to be available also for a real time application on a phone without violating the privacy of the user. Our feature list includes the following:

*(i) duration:* duration of the visit in seconds

*(ii) startHour:* time of the day when the visit started $(0, 1, \ldots, 23)$

*(iii) endHour:* time of the day when the visit ended $(0, 1, \ldots, 23)$

*(iv) nightStay:* proportion of the visit duration that is between 6 pm and 6 am

*(v) batteryAvg:* average battery level

*(vi) chargingTimeRatio:* proportion of the visit duration when the charging has been on

*(vii) sysActiveRatio:* proportion of the visit duration when the system has been active

*(viii) sysActStartsPerHour:* number of status changes from system inactive to system active divided by the visit duration in hours.

For features related to calls, both incoming and outgoing voice calls are taken into account:

*(i) callsTimeRatio:* the ratio of accumulated duration of calls to the duration of the visit

*(ii) callsPerHour:* number of calls divided by the visit duration in hours.

The features related to accelerometer based motion mode detection were computed using the reported motion modes. However, as the report for one time instance may include several different modes and includes also their probabilities, we used the probabilities to weight the times for the motion modes:

*(i) idleStillRatio:* proportion of the visit duration when the status is *idle/still*

*(ii) walkRatio:* proportion of the visit duration when the status is *walk*

*(iii) vehicleRatio:* proportion of the visit duration when the status is *car/bus/motorbike* or *train/metro/tram*

*(iv) sportRatio:* proportion of the visit duration when the status is *run, bicycle,* or *skateboard.*

In addition to these 14 calculated features, we also saved the place label and user ID to be used in the training and testing of the models:

*(i) placeLabel:* three possible labels: *Home*, *Work*, or *Other* (the last includes all the generally less frequent places, such as friend's home, transportation, and restaurant)

*(ii) userId:* each data sample includes a unique user identifier.

The MDC data includes place labels that were provided by users [9]. First, the data were collected and the relevant places for each user were clustered. In a later stage, users were shown all the places on a map and were asked to label these places. We only consider places labeled with certainty and left out those places that users were not sure about or did not label.

In total, the visits data includes 55,932 labeled visits by 114 distinct users. From the visits, 28,921 instances are to Home (52% of all visits), 21,697 instances to Work (38%), and 5,314 instances to Other places (10%).

*3.2. Data Set #2.* Data set #2 was provided by Microsoft and collected by 16 users working in the ICT field. The average time the participants collected data was 26 days and the maximum time was 64 days. The description and results on this data set have not been published earlier.

In this data set, the data was associated with places. The place was identified by its physical location, obtained, for example, from the GNSS receiver or cellular network based positioning. The first time the user visited a place, a new data entry for that place was created. Every time the user visited a once created place, the phone accumulated time counters for several status variables. The *stay time* was the accumulated time the phone was observed to be in the place and *night stay* was the accumulated time the phone was there between 6 p.m. and 6 a.m.

The accumulated times included also the times with the motion states *idle, stationary, moving, walking,* and *vehicle*, all determined by the sensors of the phone. The third group of times recorded included phone usage data: time with *display on* and *charging* times, time spent on *calling*, and time with *headset on*.

In addition to these, the *total time* since the place data entry was created was recorded. To the place data, the user-given semantic label, such as *Home* or *Work* was also associated. The physical location of the place was not included in the data. Twice a day, when a data connection was possible, the phone application sent the recorded time countervalues to a server. Thus the database included the history of the countervalues that were sampled at approximately 12 h intervals. This data set differs significantly from data set #1 in that the individual visits to a place cannot be detected or counted and neither can the individual phone calls or activity starts. From this data, we computed the features to be used in classification by dividing the time countervalues by the total time. Similarly, as with data set #1, we lumped all other user-given place labels, except *Home* and *Work*, to the third label *Other*.

In total, data set #2 includes 5,605 labeled samples by 16 distinct users. From these samples, 1,747 cases (31% of all visits) are labeled with *Home*, 1,482 with *Work* (26%), and 2,376 with *Other* (42%). Each sample consists of 11 features related to stay, activities, and phone usage and additional information such as user id, place label, and total time recorded for the location.

With both data sets #1 and #2, regarding the accelerometer based recognition of the motion state or activity, we rely on the output of the motion or activity recognition functions of the phone applications and data set providers. The reliability statistics of the functions are not known to us. For our classification functions, the possible errors in these features are noise in the data.

## 4. Methods

We consider three alternatives for the data representation: visits data representation, places data representation, and cumulative samples; these terms are explained in Section 4.1. Once the data is extracted from the database in the representation schemas, we apply seven well-known classification methods. Our goal is to determine which classification method and which data representation approach is the best for the semantic labeling of places. We also describe the cross-validation method we used to assess the performance of the classification, the sequential feature selection method used to improve the accuracy and to assess the significance of the individual features, and the approach used for assessing the confidence in the classification results.

*4.1. Data Representations.* In this paper, we consider three different approaches to represent the data. The visits approach uses the features computed for each visit as such, so that the data includes several samples of one user's visits to each of the user's places. That means that there is one tuple for each location-user-event. Therefore, a user visiting home 3 times adds three tuples to the learning data. From data set #1, we extract 55,932 labeled visits by 114 users.

The places approach combines all the visits of one user to one place into a single summarized sample. That means that there is one tuple for each location-user, which is calculated combining all the relevant visit tuples. The idea is to assume that different users tend to use their phones in similar ways in semantically similar places, for instance, at home. From data set #1, we extract 295 labeled places by 114 users. For instance, if a user visited home ten times in a week, the visit data representation creates ten different data instances,

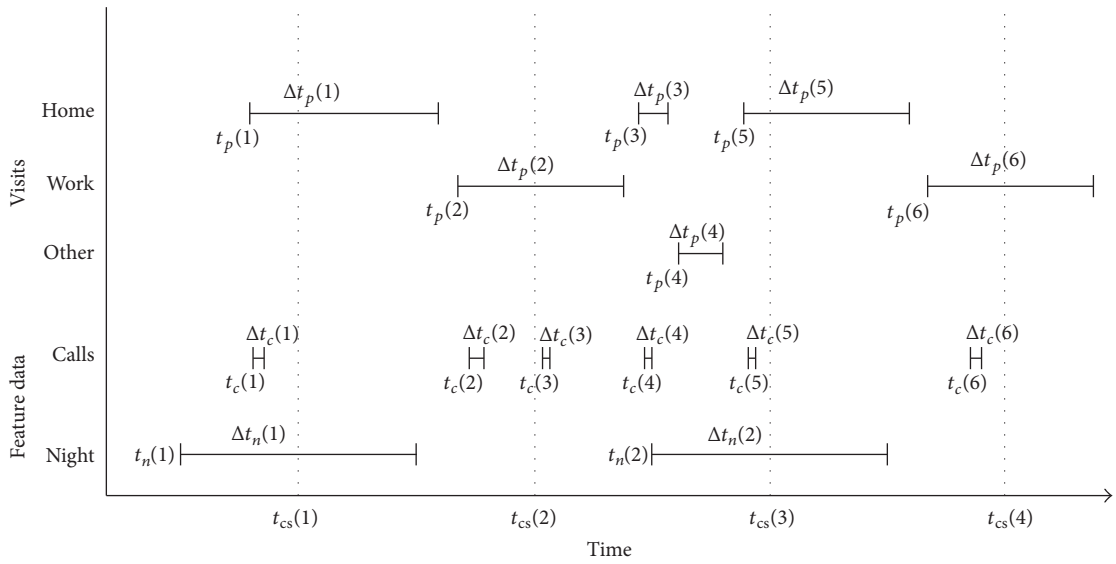



Figure 2: Example: time tags, durations, and place labels for computing features related to phone calls and night stay. The time of the $j$th cumulative sample is $t_{cs}(j)$.

while the place data representation combines the ten visit data instances into one place data instance. The visits and places representations are available only for data set #1.

The third data representation includes cumulative samples of the features. This representation is the native representation of data set #2; that is, it includes the accumulated times of staying and phone usage in a place by one user. To obtain similar samples from data set #1, we computed the accumulated times of stay, activities, and phone usage for each user-place combination. We took samples of these accumulated times at 12 h intervals and divided them by the total times since the first sample of the user-place combination. The 9 features of data set #1 that were converted to cumulative samples are the following: *stay, nightStay, charging, sysActive, calling, idleStill, walking, vehicle*, and *sport*.

Figure 2 and Algorithm 1 illustrate the computation of the different data representations for features related to phone calls and night stay. Feature *nightStay* is chosen as an example because it is computed differently from all the other features and therefore needs to be described separately. By contrast, feature *calling* is similar to all other features, and the description of its computation can be applied also to these. Figure 2 illustrates the notation for the time tags and durations. The start time and duration of visit $i$ are $t_p(i)$ and $\Delta t_p(i)$, respectively. In addition to these time attributes, the label place($i$) is attached to the visit data. Similarly, $t_c(i)$ and $\Delta t_c(i)$ represent the start time and duration of a call, and $t_n(i)$ and $\Delta t_n(i)$ are the start and duration of night. In our implementation, $\Delta t_n(i)$ is constant 12 h. We also make the simplification regarding calls that span over a visit so that the whole call is associated with the visit where it started.

Algorithm 1 presents equations for the computation of the features for the different data representations. As examples of combining data from several visits to places and cumulative samples data representations, we use *Home* as the example of user's place. The feature for number of phone calls is not included in cumulative samples and, therefore, it is not included in Algorithm 1. On the other hand, *nightStay* is included only in the feature set of cumulative samples. Although a call can take place during the night, in the data representation *nightStay* and *calling* are not directly connected. However, for the classifier model it is possible to learn the connection as their simultaneous occurrence increases both counters *(calling*, *nightStay)* simultaneously.

*4.2. Classification Methods.* In this work we apply the following classification methods using their implementation in the Statistics and Neural Networks toolboxes of Matlab.

*Naïve Bayes (NB)* [28–31] is a statistical approach having an explicit underlying probability model and it provides a probability of being in each class rather than simply a classification. Naïve Bayes assumes that features are conditionally independent; this reduces computational cost and often works well even if the independence assumption does not hold. There are no tuning parameters in this approach.

*Decision tree (DT)* [28–31] is a machine learning approach that probably gives the most understandable results by humans, who can identify the most relevant features. For attribute selection we use Gini's diversity index. The features selected at the top of the tree are the most relevant features for the classification. There are two options to avoid overfitting, prepruning, and postpruning. We chose postpruning since prepruning requires determining when to stop growing the tree while building it, which is not an easy task. When the tree is built we postprune the tree using Error Estimation. Intuitively, the method goes through the nodes of the tree comparing the original tree with the tree pruned on that node. The tree is pruned in that node if the pruned tree improves (or equals) the classification accuracy.

Assume that start times and durations are available for
(i) visits: $t_p(1), t_p(2), \ldots$ and $\Delta t_p(1), \Delta t_p(2), \ldots$
(ii) calls: $t_c(1), t_c(2), \ldots$ and $\Delta t_c(1), \Delta t_c(2), \ldots$
(iii) nights: $t_n(1), t_n(2), \ldots$ and $\Delta t_n(1), \Delta t_n(2), \ldots$
**visit** $i$ (compute for all $i$)
(i) find the smallest index $k_1$ such that $t_p(i) \leq t_c(k_1)$
(ii) find the largest index $k_2$ such that $t_c(k_2) \leq t_p(i) + \Delta t_p(i)$

$$f_{\text{callsTimeRatio}}(i) = \sum_{k=k_1}^{k_2} \frac{\Delta t_c(k)}{\Delta t_p(i)}$$

$$f_{\text{callsPerHour}}(i) = \frac{(k_2 - k_1 + 1)}{(\Delta t_p(i) c_{t2h})},$$

where multiplication with $c_{t2h}$ converts the time units to hours
**places: home**
(i) find set $H$ of all visit indices $i$ such that **place**$(i) = Home$

$$f^H_{\text{callsTimeRatio}} = \frac{(\sum_{i \in H} f_{\text{callsTimeRatio}}(i) \Delta t_p(i))}{(\sum_{i \in H} \Delta t_p(i))}$$

$$f^H_{\text{callsPerHour}} = \frac{(\sum_{i \in H} f_{\text{callsPerHour}}(i) \Delta t_p(i))}{(\sum_{i \in H} \Delta t_p(i))}$$

**cumulative sample** $j$: **home** (compute for all $j$)
Computation of $f^H_{\text{calling}}(j)$ for calling at home:
(i) find set $H$ of all visit indices $i$ such that **place**$(i) = Home$
and $t_p(i) + \Delta t_p(i) \leq t_{\text{cs}}(j)$

$$f^H_{\text{calling}}(j) = \frac{(\sum_{i \in H} f_{\text{callsTimeRatio}}(i) \Delta t_p(i))}{(t_{\text{cs}}(j) - (t_p(\min_{i \in H} i)))}$$

Computation of $f^H_{\text{nightStay}}(j)$ for night stay at home:
$a = 0$
**for** all $i \in H$
**if** exists $k$ such that $t_n(k) \leq t_p(i) \leq t_n(k) + \Delta t_n(k)$

$$a = a + \min(t_p(i) + \Delta t_p(i), t_n(k) + \Delta t_n(k))$$

**else if** exists $k$ such that $t_n(k) \leq t_p(i) + \Delta t_p(i) \leq t_n(k) + \Delta t_n(k)$

$$a = a + t_p(i) + \Delta t_p(i) - t_n(k)$$

**end for**

$$f^H_{\text{nightStay}}(j) = \frac{a}{(t_{\text{cs}}(j) - (t_p(\min_{i \in H} i)))}$$

Algorithm 1: Example: computing features related to phone calls and night stay in different data representations.

*Bagged tree (BT)* [29–32] combines different decision trees (with the same parameters as the decision tree above), each of which has been trained using different portions of the data. Using a voting system, each tree is given more weight in the region of the space where its classification rate is better. This method is proved to work better than single decision trees. We use ten decision trees, a typical value.

*Neural network (NN)* [28–31, 33] is a brain-physiology inspired classifier. It consists of layers of interconnected nodes, each node producing a nonlinear function of its input. The input to a node may come from other nodes or directly from the input data. Some nodes are identified with the output of the network. In particular, we used a multilayer perceptron with one hidden layer that contains ten hidden neurons. The decision of having these settings is based on the limited number of samples and the authors' experience. To train the network we used Levenberg-Marquardt optimization to update the weight and bias values. Neural network for classification assumes that the class labels are represented as binary vectors. Therefore, before training the class labels are coded as vectors: $Home \rightarrow [1, 0, 0]$, $Work \rightarrow [0, 1, 0]$, and $Other \rightarrow [0, 0, 1]$. The neural network predictions are also vectors. However, their element values are not exactly ones and zeros. The predicted classes are obtained by finding the index to the largest element of the output vectors and converting these back to class labels.

*K-nearest neighbours (KNN)* [28–32] is a statistical method that classifies an incoming instance according to the distance to the $k$-nearest points in the training set. We used Euclidean distance to choose the nearest neighbours. We determined the values of $k$ to be used in classification using leave-one-user-out validation and classification accuracy as optimization criterion (see Section 4.4). We found that the best $k$ value depends on data set and data representation: with data set #1 the best $k$ values were 27, 3, and 57 for visits, places, and cumulative samples, respectively. With data set #2 the best accuracy was obtained with $k = 159$. For large training data sets, the required storage for the model is large, and

Table 1: Missing MDC data instances.

| Data representation | Partial system data | Accelerometer based data | Both |
|---|---|---|---|
| Visits | 192 | 36,543 | 25 |
| Places | 21 | 6 | 0 |
| Cumulative samples | 3,903 | 41,299 | 1,513 |

also the CPU time to find the nearest neighbours gets large. This may be prohibitive for applications running on resource constrained mobile devices.

*Support vector machine (SVM)* [30, 31, 33] is a binary classifier; that is, it can be applied for classification problems with two classes. A SVM seeks a hyperplane that best separates the features of one class from the features of another class. Its goal is to find a hyperplane that maximizes the zone on both sides of the hyperplane such that the zone does not include feature vector samples. The feature vectors closest to the found hyperplane are called support vectors. In many problems the separation of the classes cannot be done using a simple hyperplane. Therefore, the method includes a possibility of using linear or nonlinear kernel functions to produce a hypersurface that performs the separation. We used Gaussian Radial Basis Function (RBF) as the kernel function. With our data, we obtained similar accuracy with both the RBF and the linear kernel functions but RBF required smaller number of support vectors. We used Matlab's `fitcsvm` function to train the SVM classifiers. To set the RBF sigma parameter we used `KernelScale=1` which we found to work best with the data when compared with several other `KernelScale` values. The solution for our 3-class problem was obtained by using 3 binary classifiers to provide one-versus-all other classifications: *Home* versus *Not Home*, *Work* versus *No Work*, and *Other* versus *No Other*. For the binary classifiers, the multiclass labels were transformed before the training as follows: (1) *Home* $\rightarrow$ 1, *Work*, or *Other* $\rightarrow$ 0; (2) *Work* $\rightarrow$ 1, *Home*, or *Other* $\rightarrow$ 0; (3) *Other* $\rightarrow$ 1, *Home*, or *Work* $\rightarrow$ 0. In the prediction phase, the binary classifiers were used to obtain the posterior probabilities of their active class. The binary classifier with the largest posterior probability was used to determine the multiclass output.

*Logistic regression (LR)* [28, 31] models present the probability of the class as a logistic function of a linear regression expression of the features (linear combination of the features and a constant). LR is also a binary classification method. Therefore, we made a transformation of multiclass labels to several binary classes as we did in the case of SVM and trained three LR models. In the prediction phase the three classifiers were used to obtain the probabilities of the classes, and the class with the largest probability were chosen as the multiclass output. However, sometimes the linear regression problem is ill-conditioned and regularization is needed in order to obtain the parameter estimates. We used Lasso regularization for generalized linear model regression and constructed a regularized binomial regression model with 4 different values for regularization parameter $\lambda$ and 2-fold cross validation. With these values the time consumed in parameter estimation remained moderate and the obtained model parameters provided good prediction accuracy.

*4.3. Missing Data.* With the MDC data, we encountered a problem with missing data. The data includes visits where either the system data partially (i.e., features *batteryAvg, chargingTimeRatio, sysActiveRatio,* and *sysActStartsPerHour*), the acceleration based activity data in full, or both of these data are missing. As the places and the cumulative samples representations are computed from the visits data, these representations inherit the problem. The numbers of instances with missing data in each of the data representations are shown in Table 1.

The instances with missing data cause problems in the training of the LR model and degrade the performance of other classifiers, especially NB, NN, and KNN. To mitigate the effect of missing data, we trained four variants for each classifier: the first one uses all the features; the second one uses all other features except the sometimes missing system features; the third one uses all other features except acceleration based ones, and the last one uses neither the sometimes missing system features nor the acceleration based features. The classifier variants were trained using only samples where all the features used by the classifier were available. In the evaluation of the classifier, the decision on which classifier variant to use for classification was made separately for each test data sample, we chose the classifier variant that did not require the features that were missing in the sample but used as many as possible of the features available.

*4.4. Performance Evaluation of Classifiers.* Once we have built the classifiers based on the training data, we use the test data to evaluate the classifiers. In machine learning, it is common to choose a certain proportion, for example, one-third, of data to a test set, which will be used only to evaluate the classifier, not to build the classifier model [29, 31]. The test set is also labeled. Therefore, we have the information about the true label (the user-given values) of the samples. In the evaluation of the classifiers, each test data sample is fed to the classifier and the output of the classifier, that is, the predicted label, is compared with the true label. Accuracy value of 53% means that 53% of the predicted values are equal to the true value; we use classification rate as a synonym of accuracy.

Our goal is to classify the data of one user by using a model based on the data of the other users; that is, we want to learn patterns that are common to all users. Therefore, splitting of the data to training and test sets is based on user id. As a result, the data of a user is not classified with the knowledge of the user's own data. Using knowledge of

future data of the user would be unrealistic, and using the knowledge of the past data of the user is a different problem, not addressed in this paper.

One option would be to randomly choose a certain proportion of users to test data. However, there is large variation in the numbers of samples by different users and the numbers of visits to each of the labeled places also differ by users. Because of this, the overall accuracy evaluations of the classifiers vary significantly depending on which users are in the test set. We solve this problem by using leave-one-user-out validation. For $n$ users, the training and testing are repeated $n$ times each time with one user's data as the test set and the remaining users' data as the training set. The overall accuracy obtained by combining the results from all the tests is used as evaluation criterion. This approach to cross-validation is deterministic, which makes the results easier to interpret when comparing several different setups, for example, in feature selection. In these comparisons, we want the variations in classifier designs, for example, the feature combination, to be the major sources of performance differences, not the random selection of test sets. The combined results include test results obtained using classifiers trained with all different training sets. It includes one classification result for each labeled data sample of each user. Note that we do not control the random initializations of training methods, which also makes some contribution to the observed differences. However, using leave-one-user-out validation and combining results of $n$ tests mitigate also the biases caused by the random initializations.

*4.5. Feature Selection.* By selecting only a subset of the available features, the number of inputs presented to the classifier can be reduced. This benefits the classification task in several aspects: fewer features result in fewer model parameters, which improves the model's ability to generalize and reduce model complexity and the run time of the algorithm. It also provides insight into the problem by distinguishing the more significant features from the less important ones [32]. Some of the learning methods such as decision trees, bagged trees, and regularized logistic regression include feature selection as an integral part of the learning procedures [31]. However, others do not. Therefore, we search for the improved feature subset by selecting candidate subsets and evaluating their predictive accuracy using the leave-one-user-out validation described in Section 4.4.

One option for selecting subsets would be an exhaustive evaluation of all the possible subsets. However, for 11 features the number of subsets to be evaluated would be 2047 and for 14 features it would be 16,383. These would require too long computation time especially with slower methods, such as SVM, when applied for testing with leave-one-user-out validation. Therefore, a search strategy is needed for selecting candidate subsets for evaluation. We apply sequential selection algorithm for this purpose.

In sequential feature selection (SFS) features are added or removed one at a time [32]. The SFS provides a suboptimal solution to the feature selection problem as it easily becomes trapped to a local minimum. To mitigate this problem, we implemented the algorithm in both forward and backward directions. SFS in forward direction is a greedy search algorithm. It adds features one by one to the model until the addition of more features does not improve the predictive accuracy any more. In backward direction, the process is started from the model, including all the available features, and then features are removed one at a time until removing features does not improve the performance. Before the decision is made on which feature is added or removed, the effect of each available candidate feature for addition or removal is tested. The candidate feature that produces the largest improvement to the predictive accuracy when compared to the selected feature set from the previous trial cycle is added or removed, depending on the direction of the search. The process ends when none of the candidates in the entire trial cycle is able to improve the performance obtained in the previous trial cycle. If the predictive accuracy is the same as in the previous trial cycle, the candidate set with fewer features is selected.

*4.6. Confidence of Classification.* In many practical classification problems, it would be useful if, in addition to providing the classification result, the classifier was also able to provide information about the quality of its classification [34]. In particular, we focus on the confidence of the classification, assessing how reliable the classifier itself considers its own decision. High classification confidence means that the classifier is "sure" about its output while low confidence means it is "unsure." The idea in the confidence assessment is to use the information about the execution of the classifier on a specific input sample to infer the confidence that the classification result generated for the sample is correct [35].

NB and LR classifiers base their decisions on the probability models of the classes, and their output is the posterior probability of the class given the feature values. These probabilities can be considered as confidence measures of the classifier outputs. The SVM produces scores as class likelihood measures and Matlab provides `fitPosterior` function to transform these to posterior probabilities. The predicted outputs of the NN based classifier are binary vectors $z$ of length $n = 3$, that is, the number of possible classes $y$. Ideally, the value of the element corresponding to the predicted class is 1 while the others are zeros. In practice, due to imperfect training examples, noise, and other mismodeled effects, the predicted elements $z_y$ are seldom exactly ones and zeros. Therefore, the classification result $\hat{y}$ is determined using the element closest to one; that is,

$$\hat{y} = \underset{y}{\operatorname{argmin}}\, d_y, \tag{1}$$

where $d_y = |1 - z_y|$. Now the distance $d_y$ serves as an indicator on how well the current feature vector fits to the NN model of class $y$. To get this value to the same scale with the probability outputs of NB, LR, and SVM, we convert the distance $d_{\hat{y}}$ to the confidence measure $c$. However, it may happen that, in some cases, when the fit of the input sample to the model is exceptionally poor, even the shortest distance $d_{\hat{y}}$ may be larger than one. Therefore, the confidence is obtained from the distance using $c = 1 - \min(1, d_{\hat{y}})$.

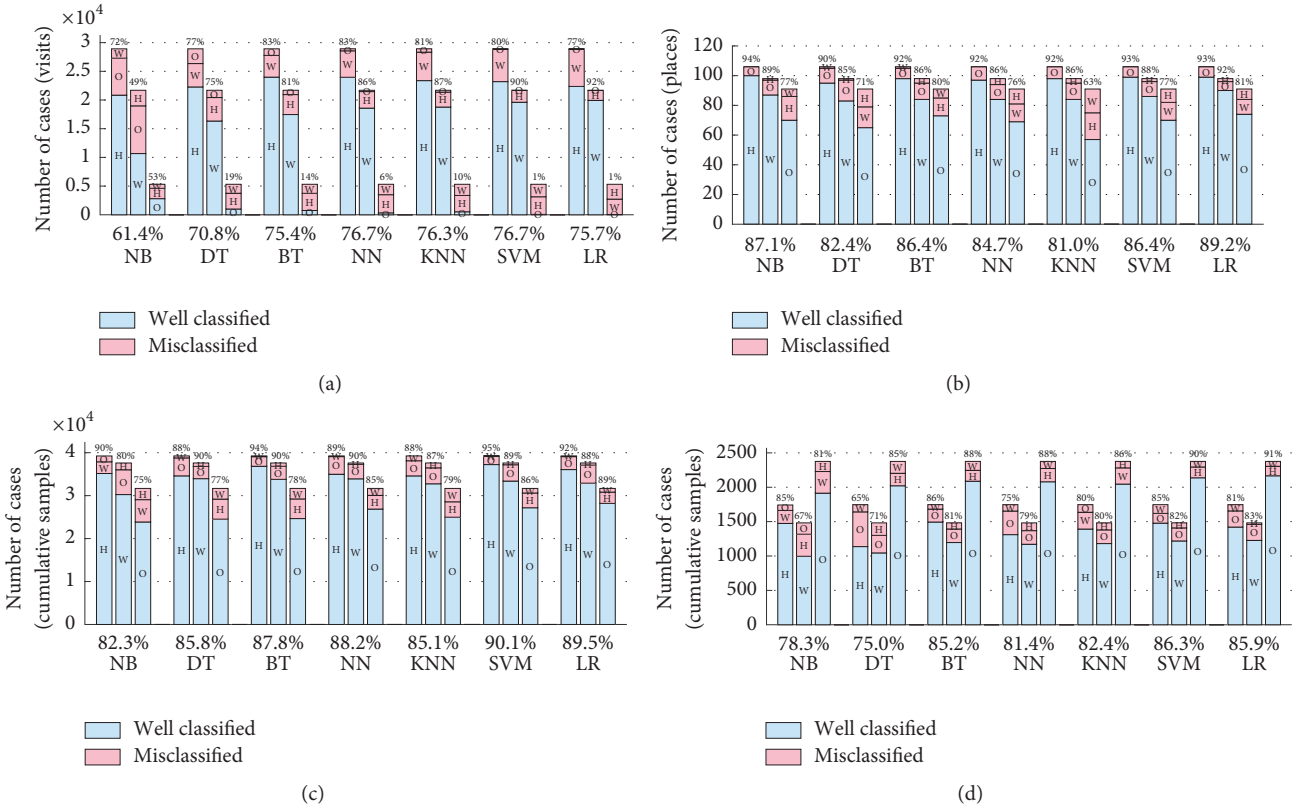



Figure 3: Classification rates (%) for different methods, using data set #1 and (a) visits, (b) places, and (c) cumulative samples approaches and (d) data set #2 and cumulative samples. The percentage of well-classified samples, that is, cases where the classification result is correct, for each class is given above the bars. The overall percentage of well-classified samples for the classifiers is shown below the bars.

Using both data sets, we study how well the confidence measure can predict the misclassification rate, that is, how well the classifier assesses its own performance. We set a threshold to the confidence, below which we say the confidence is low and above which it is high. There are four possible combinations of this measure (the confidence assessed by the classifier) and classification success: (1) well classified with high confidence, (2) well classified with low confidence, (3) misclassified with high confidence, and (4) misclassified with low confidence. The classifier produces the predicted label and confidence assessment based on the input features but without knowing the correct label. Therefore, it is possible that the classifier has high confidence but when its prediction is compared with the correct label, it turns out that the input was misclassified. We consider decisions 1 and 4 correct, as in these cases the confidence of the classifier predicts the success of the classifier, while in cases 2 and 3 the decisions are wrong as the confidence of the classifier gives wrong prediction about the success. Assuming that the costs of the unsuccessful cases 2 and 3 are equal, as well as the rewards of the successful cases 1 and 4 are equal, we search for a confidence threshold such that the ratio between the number of cases 1 and 4 over cases 2 and 3 is maximized. We use the obtained threshold to reject samples that have confidence lower than the threshold and record how much the overall accuracy of a classifier improves using the threshold and how large proportion of samples is rejected.

## 5. Results

In this section we describe our results on the comparisons of the data presentations and classification methods using the methods described in Section 4. In all the tests based on data set #1, the missing feature values in the input samples were treated as described in Section 4.3.

*5.1. Classification.* The results on the comparisons of the data representations and different classification methods are shown in Figure 3 where the evaluation criterion is the overall predictive accuracy observed in leave-one-user-out validation described in Section 4.4. The results are summarized in Table 2.

Figure 3(a) shows the classification of each method using the visits representation. All the methods but the Naïve Bayes show a certain bias. They achieve high accuracy for the places *Home* and *Work* and low accuracy for the place *Others*. The intuitive reason is that visits to *Home* or *Work* are more frequent than visits to places labeled as *Others*. Therefore, the algorithms sacrifice accuracy in *Others* to achieve higher accuracies in *Home* or *Work*.

Table 2: Accuracy results of the data representations: summary over all implemented classifiers.

| | Data representation | | | |
|---|---|---|---|---|
| | a | b | c | d |
| min | 61.4 | 81.0 | 82.3 | 75.0 |
| max | 76.9 | 89.2 | 90.1 | 86.3 |
| mean | 73.3 | 85.3 | 87.0 | 82.1 |
| std | 5.7 | 2.8 | 2.7 | 4.2 |
| max-min | 15.5 | 8.2 | 7.8 | 11.3 |

Figure 3(b) shows the corresponding results using the places representation. Compared to the visits representation, the classification accuracies are higher. Also, the differences between the accuracies of the classifiers are smaller than with the visits approach. The improvement obtained by combining of all the visits to one place may be because generally averaging reduces the effect of the outliers. The disadvantages of the places representation are the following. First, it is more computationally expensive to produce because of the need to combine all the individual visits to places. The second disadvantage is the so-called cold start problem: the classification algorithm will not classify accurately the places until a certain number of visits to a place have been collected.

The classification results with cumulative samples of data sets #1 and #2 are shown in Figures 3(c) and 3(d). The cumulative samples with data set #1 improve the accuracy and decrease the differences between the classifiers even more than the places approach. Cumulative samples include averaging similarly as the places representation and the generation of cumulative samples reduce variability in samples if the phone usage and place visiting pattern stay regular. However, the computation of cumulative samples also generates some variability, as it produces samples even when new visits have not been made to the place. In this case the feature values change as the total time used for scaling still grows even though the cumulative times of the stay and activities remain constant. The averaging together with the much larger number of samples provides a plausible explanation to the improvement. With the cumulative samples of data set #2, the accuracies are lower and the accuracy differences between the classifiers are larger. This could be due to the smaller size and time span of the data.

When comparing the results of different classifiers with all the data representations, SVM and LR are always among the three algorithms that provide the best classification accuracies while DT is among the three classifiers with the worst accuracy. BT and NN also perform quite well; they are never in the group of the worst three. Generally NB does not provide good accuracy, except that with the places representation it is the second in accuracy. From the classifiers studied in this paper, SVM is by far the slowest classifier to train. The classification with the trained SVM is fast; however, its memory requirements in classification phase become high if the number of support vectors is high. The issue is emphasized in multiclass classification as the support vectors need to be stored for each class separately. Therefore, despite its accuracy, SVM mainly serves as a reference, and we do not consider it to be suitable for practical applications with this type and amount of data in resource constrained mobile devices. The computational cost in prediction is also high with KNN as it has to store all the training samples and compare them with the new input. Therefore, its practical applications are restricted to cases where extreme simplicity of the algorithm is required but high computational costs can be accepted. Based on these comparisons, LR, NN, and BT seem to be the most promising methods for practical applications.

Our test results indicate that data representations including averaging, that is, places and cumulative samples, give higher classification accuracies than visits data representation. The average classification accuracies with visits, places and cumulative samples obtained from data set #1 were 0.72, 0.85, and 0.87, respectively, and 0.81 with the cumulative samples of data set #2.

*5.2. Effect of Accumulation Time with Cumulative Samples.* With the cumulative samples, the samples themselves evolve in time as new data are accumulated to the time counters of the features. To study the effect of the accumulation time to the classification accuracy, we grouped the samples based on the accumulation time $t_{\text{acc}}$. The first group included the samples where $t_{\text{acc}} \leq 1$ day, in the second group, was the samples with $t_{\text{acc}} \leq 2$ days and so on, until 7 days. These seven groups include the samples from the first week the user starts to visit a place. Into the eighth group we included all the samples, which gives the same classification accuracy that is illustrated in Figures 3(c) and 3(d).

In the training of the classifiers we used all the cumulative samples of all other users, so that the time based selection of samples did not affect the training phase. The results for the cumulative samples representation of data sets #1 and #2 are shown in Figures 4 and 5, respectively. In the figures, in addition to the overall classification accuracies, also the classification accuracies of the specific labels (*Home*, *Work*, and *Other*) are shown.

Comparing the overall classification accuracy of cumulative samples in Figure 4(a) and visits representation in Figure 3(a), it can be observed that after 6 days of accumulation time, the accuracy with cumulative samples is equal to or better than the accuracy with visits with all classifiers except NN and KNN. With these two the accuracy with visits were 76.7% and 76.3% while with cumulative samples and 6 days of accumulation time the accuracies are only about 72%. In Figure 4, the curves corresponding to the overall

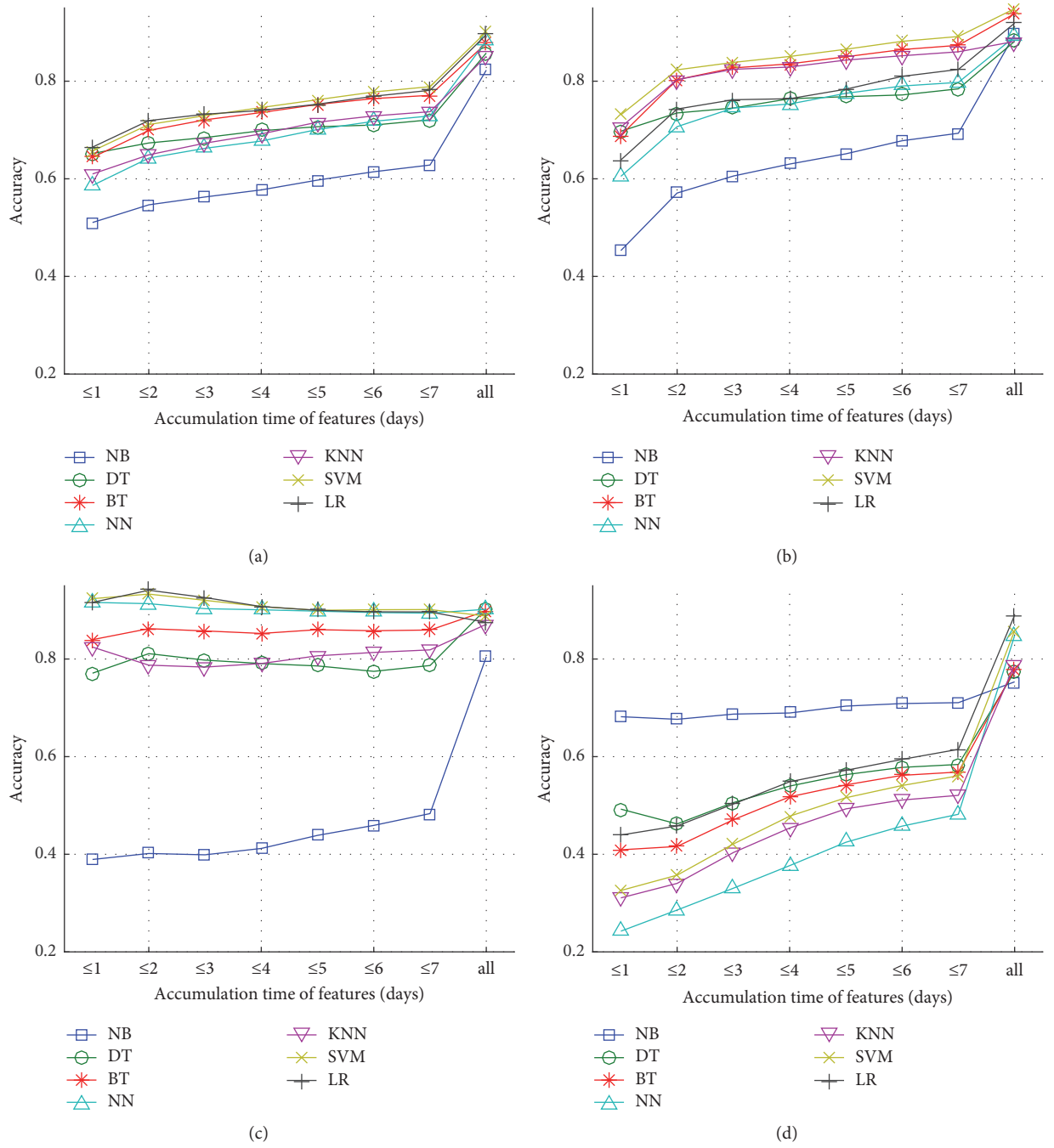



Figure 4: Effect of the accumulation time to the classification accuracy with cumulative samples of data set #1: (a) overall accuracy and classification accuracy of (b) Home, (c) Work, and (d) Other places. Accuracy as ratio (unitless).

accuracy and classification accuracy of *Home* and *Other* for all the classifiers are monotonically rising after 2 days; that is, the accuracy improves as the accumulation time of feature samples increases. There is also clear improvement from 7 days to the maximum accumulation time. The classification accuracy of *Work* behaves differently: with all classifiers except NB the rise of the accuracy is very slow and it is not monotonically rising.

Based on these results, for *Work* gathering more information by integrating the values for longer time does not improve its accuracy as happens with *Home* and *Other*. The data sets differ in that with #1 there is clear accuracy improvement when accumulation time increases from 7 days, while with #2 there is no clear improvement; with *Home* even a decrease of the accuracy can be observed. This is probably due to the smaller total number of samples and

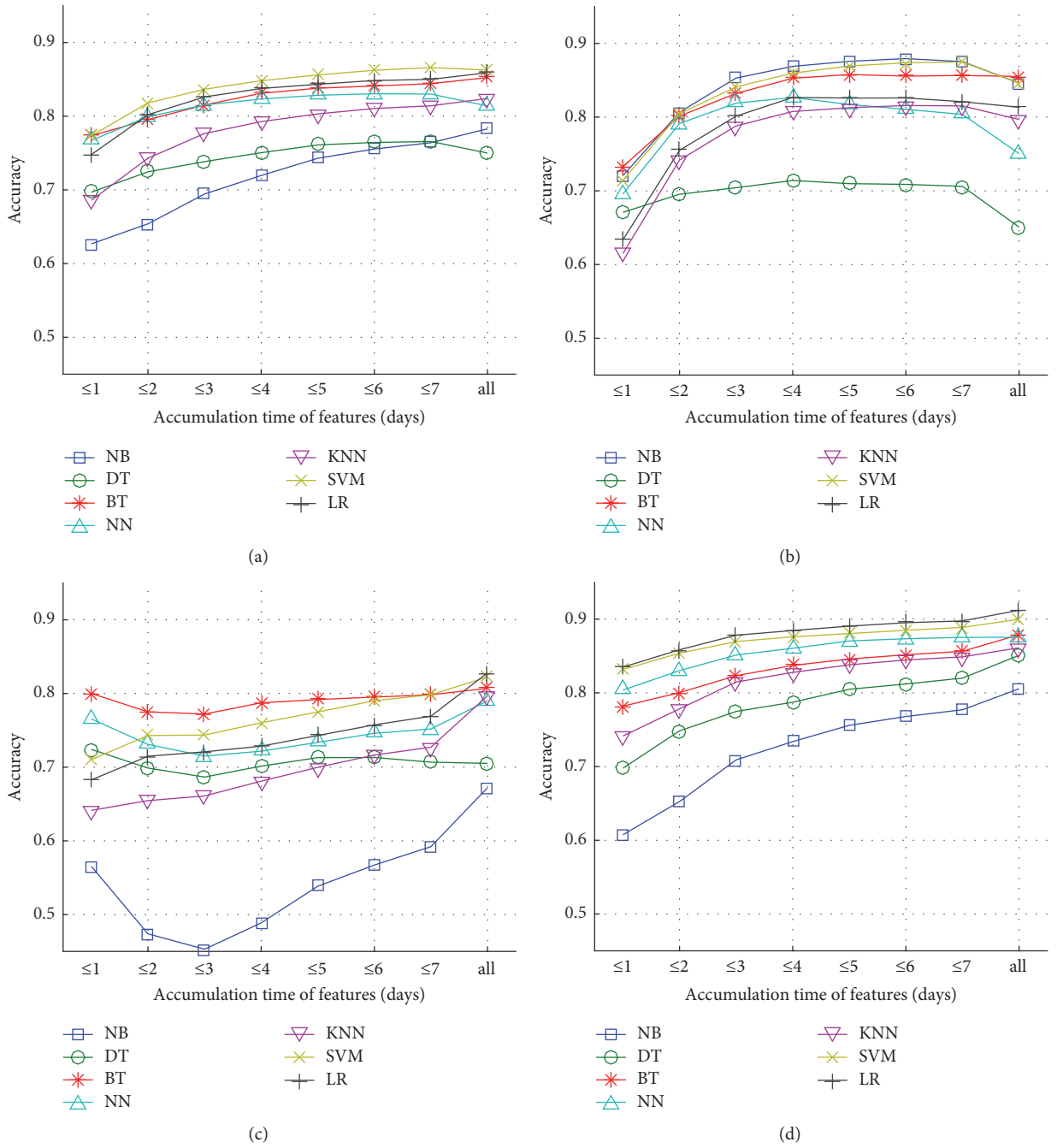



Figure 5: Effect of the accumulation time to the classification accuracy with cumulative samples of data set #2: (a) overall accuracy and classification accuracy of (b) Home, (c) Work, and (d) Other places. Accuracy as ratio (unitless).

shorter data collection times in data set #2. The histograms of accumulation times of the samples in both data sets are shown in Figure 6. With data set #2, about half of the samples have accumulation time less than 7 days. With longer accumulation times, the data is biased by only few users, which reduces the reliability of results on longer accumulation times.

In Figure 5(a) the overall accuracy approaches the final accuracy already after 4-5 days accumulation: only NB improves significantly; after that, BT, KNN, and LR improve only slightly, and the accuracies of DT and NN decrease. Comparing different data sets, the cumulative samples in Figure 5(a), and the visits representation of data set #1 in Figure 3(a), it can be seen that already after 2 days of data accumulation the accuracies with cumulative samples exceed the accuracies of visits. In Figure 5, only the classification accuracy of *Other* is monotonically rising for all the classifiers. In the accuracy of *Home* there is a clear drop from 7 days to the maximum accumulation time with all classifiers except BT, and with DT, NN, and LR the decrease starts even earlier

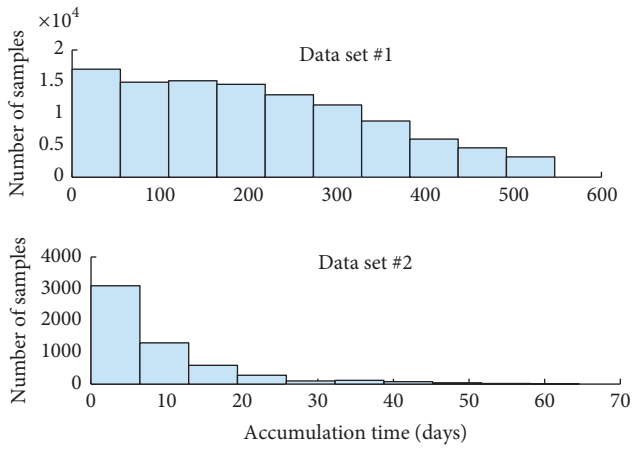



Figure 6: Histograms of the accumulation times with both data sets.

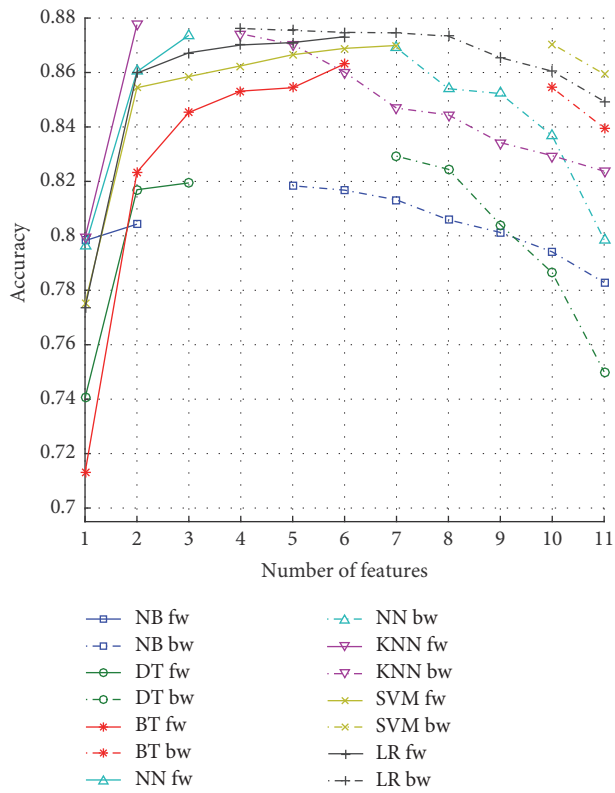



Figure 7: Sequential feature selection with several classification methods, in forward and backward direction. Accuracy as ratio (unitless).

before accumulation time of 7 days. In the classification of *Work* the accuracy behavior differs from other classes: with all other classifiers except SVM and LR, the accuracy first decreases with accumulation time and then starts to increase. With DT, the final accuracy is even worse than in the beginning. However, the increase of accuracy is very slow, except with NB. In spite of these effects in the classification accuracy of individual classes, in the first 7 days the overall accuracies shown in Figure 5(a) increase as the accumulation time increases. However, the accuracy with the maximum accumulation time with DT and NN is smaller than with 7 days of accumulation.

Generally, the longer time the data has been accumulated, the more accurately the data sample will be classified. The average accuracy obtained using the visits representation of data set #1 is exceeded by cumulative samples of data set #1 after 6 days of accumulation while with data set #2 that happens already after 2 days of accumulation.

*5.3. Feature Selection.* Sequential feature selection (SFS) in both forward and backward directions for all the classifiers described in Section 4.2 was applied to data set #2. The results are shown in Figure 7, where the overall accuracy of the classifier is shown as a function of the number of features. The curves with solid line show the results of SFS in the forward direction. For each classifier, the line starts from the left with one feature and continues until the addition of new features does not improve the accuracy any more. The results of SFS in the backward direction are shown with dash-dot lines. These curves start from the right with all 11 features included and continue to the left decreasing the number of features until removing features does not improve the results any more. The accuracies with just one optimally chosen feature are between 0.69 and 0.79 while with all features the accuracies are between 0.74 and 0.86. The accuracies using the best feature subsets found with forward and backward algorithms are between 0.82 and 0.87. Thus the selection of the features decreases the accuracy differences between the classifiers.

With NN, BT, and KNN, the forward selection yields better accuracy than backward selection and the number of selected features is also smaller. With NB, DT, and LR, the obtained accuracy in backward direction is better. With LR, the number of selected features in the backward direction is also smaller than in the forward direction, while with NB and DT better accuracy is obtained using more features than those selected by forward SFS. Using SVM, the best accuracies in both directions are approximately the same. However, in the forward direction only 7 features are needed while in the backward direction 10 features are required for the same accuracy. The three best accuracies are obtained using LR with 4 features, NN with 3 features, and SVM with 7 features. Interestingly, the accuracy using NN with just one optimally selected feature is approximately the same as with NN with all 11 features included.

The evolution of the feature subset composition during the forward and backward SFS is shown in Figure 8. The features selected in forward selection are shown in Figure 8(a): the bigger the weight and the size of the squares were, the earlier the corresponding feature was selected. The features that did not get selected at all are not marked with squares. Figure 8(b) shows the feature removals performed in the backward selection. The large dark squares show the features that were not removed during the selection process. The smaller and lighter the square was, the earlier the feature was removed; if the size and weight are reduced, the feature is not included in the final subset. Note that, in Figure 8, the

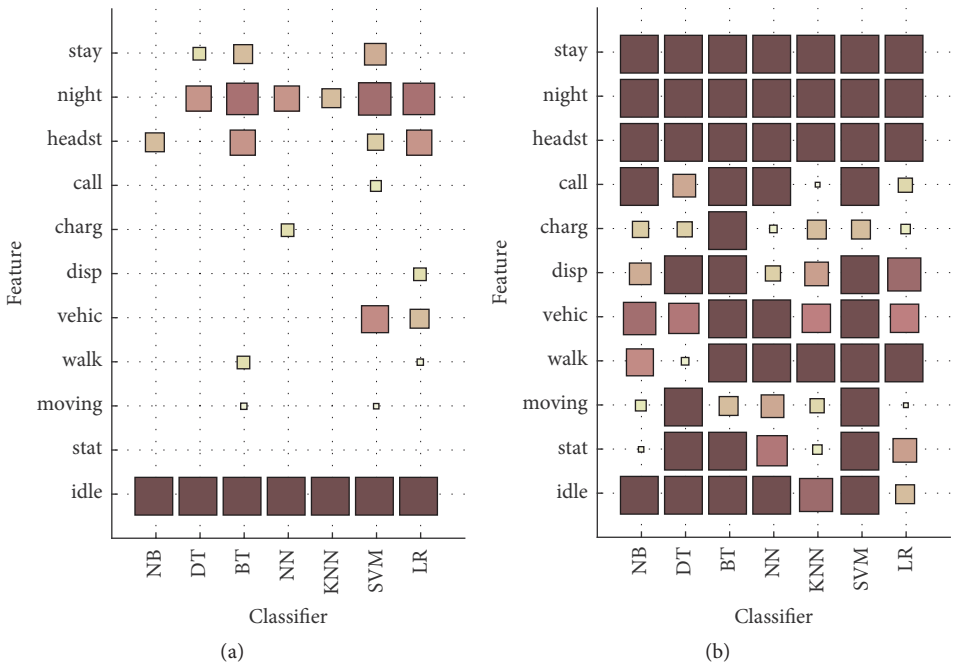



Figure 8: Importance of the features, based on sequential feature selection in (a) forward direction and (b) backward direction.

size and weight scales of the squares are comparable only within classifiers with the same final number of features. No feature is included in all final subsets, covering all classifiers and both directions of SFS. In forward direction, *idle* was the first feature selected into the model with all classifiers, and *nightStay* was the second feature selected with all other classifiers except NB. In backward direction, *stay*, *nightStay*, and *headSet* are included in all the final subsets and *idle* is included in final subsets of all classifiers except LR.

Based on these tests, we see that, even with the same training and test sets, the relevance of the features depends on the classifier. However, features *stay*, *nightStay*, *headSet*, and *idle* seem to be relevant for most of the classifiers. The selected feature sets provided improvements to the overall accuracy in the range 0.02–0.07, resulting in accuracies in the range 0.82–0.88. It can be noted that the accuracy of also the classifier models that inherently perform feature selection or extraction in their training phase, that is, DT, BT, and LR in our tests, can be improved using external feature selection algorithm. However, the results in feature importance are considered only as preliminary, as the small size of data set #2 reduces the reliability of these results.

With NB, DT, and SVM, the subset selected in forward direction is included into the final subset obtained in backward selection. With BT and NN, the features that were last selected in forward direction were first removed in backward direction and with LR the feature that was first selected in forward direction was the fourth feature removed in backward direction. This suggests that with this data, combining both the forward and backward selection in the SFS algorithm could improve the selected feature subset when accuracy is used as the selection criterion.

*5.4. Confidence of Classification.* To evaluate the relation between accuracy and the confidence measures defined in Section 4.6, we collected all the classification results and their confidence values that were obtained using test data and NB, NN, SVM, and LR classifiers. We ordered the results based on the confidence measure and divided them into 20 equal sized groups. For each of the confidence groups, we calculated the overall classification accuracies. The accuracies of these groups are shown in Figure 9 for both data sets and all the data representations.

With all the data representations, it is clear that the accuracy is significantly lower in groups with lower confidence value. However, even these groups include also well-classified samples. In the results in Figure 9(b), obtained with data set #1 and the places approach, the curves include many spikes. This is a quantization effect due to the small total number of samples. In general the curves in Figure 9(a) are smoother than in Figures 9(b)–9(d). Also the curves in Figure 9(a) show a more steady rise when compared to curves in Figures 9(b)–9(d) which present saturation-like behavior. One possible reason to the difference is the filtering that has been applied to the samples in Figure 9(b) by averaging the visits data and in Figures 9(c) and 9(d) by integrating the raw data.

In Figure 10 the ratios between the correct and wrong decisions of the classifiers are shown as a function of the confidence threshold. The threshold was used to reject classification results with confidence lower than the threshold. Correct decisions included the cases where the sample was classified correctly with confidence equal to or higher than the threshold or it was misclassified with confidence lower

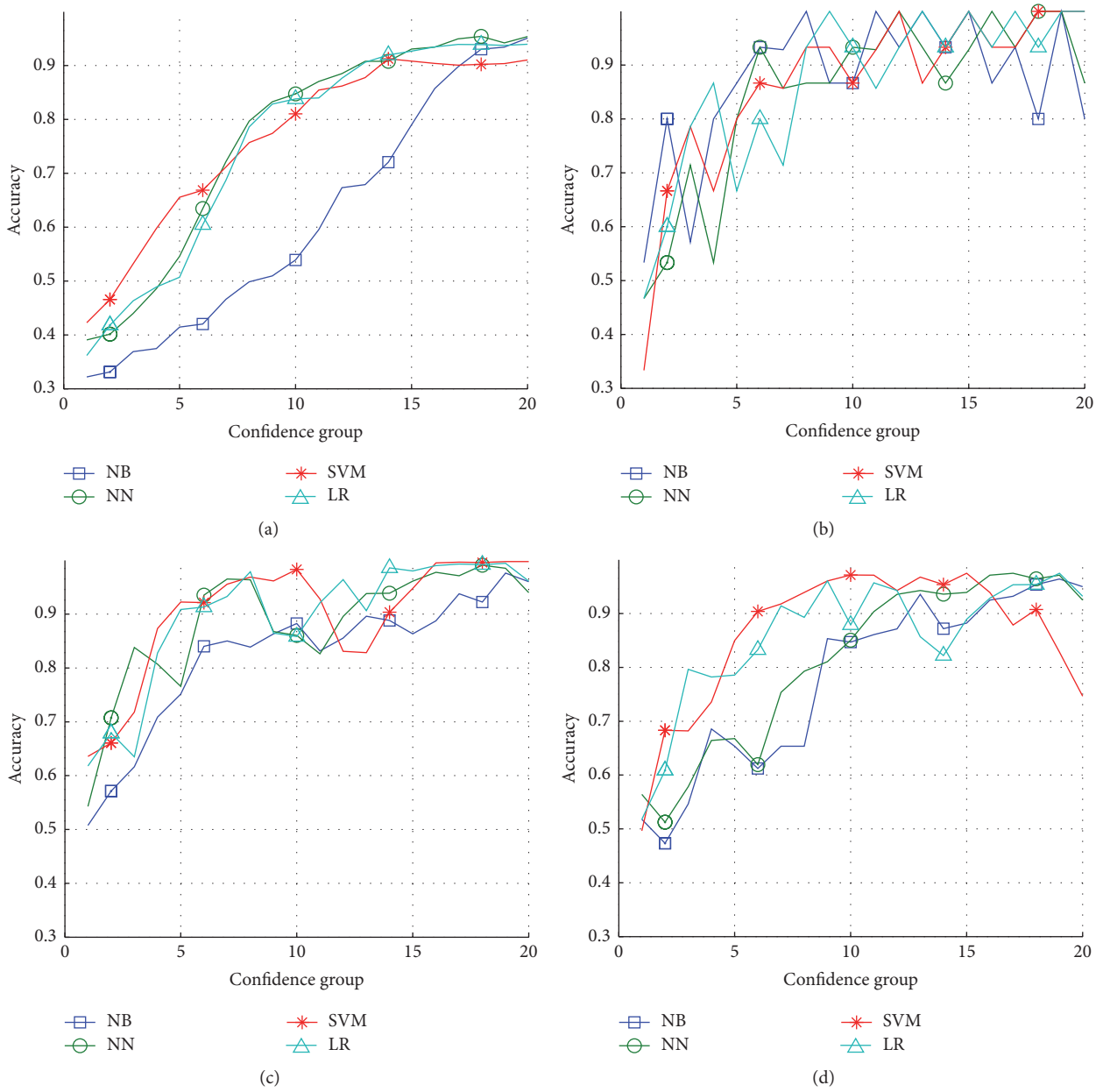



Figure 9: Accuracy (as ratio, unitless) in sample groups based on increasing confidence of classification for different data representations: data set #1, (a) visits, (b) places, and (c) cumulative samples; (d) data set #2, cumulative samples.

than the threshold. Wrong decisions included the cases well-classified with low confidence or misclassified with high confidence. The curves in Figure 10(a) are concave and smooth and include also parts where the curve is rising, making it easy to find maximums in the middle parts of the curves. In Figures 10(b) and 10(d) there are no clearly rising parts in the curves and in Figure 10(b) the curves are again wrinkled similarly as in Figure 9(b). The curves of LR in Figures 10(c) and 10(d) and SVM in Figure 10(c) are monotonically decreasing; that is, they have their maximums with the smallest confidence threshold.

Figure 11 illustrates the effect of the confidence threshold that maximizes the ratio between the numbers of correct and wrong decisions when the threshold is used to reject classification results with low confidence. Shown in the figures are the values of the confidence thresholds, the proportion of the samples rejected based on the threshold to the number of all samples, the absolute improvement of the predictive accuracy obtained by using the threshold, and the classification accuracy within the samples that are not rejected. In Figure 11(a) presenting the results of data set #1 and visits data representation, the rejection of results with lower confidence produce accuracy improvements varying between 0.05 and 0.14. With data set #1 and places data representation, shown in Figure 11(b), the improvements are clearly smaller, varying between 0.01 and 0.03. With

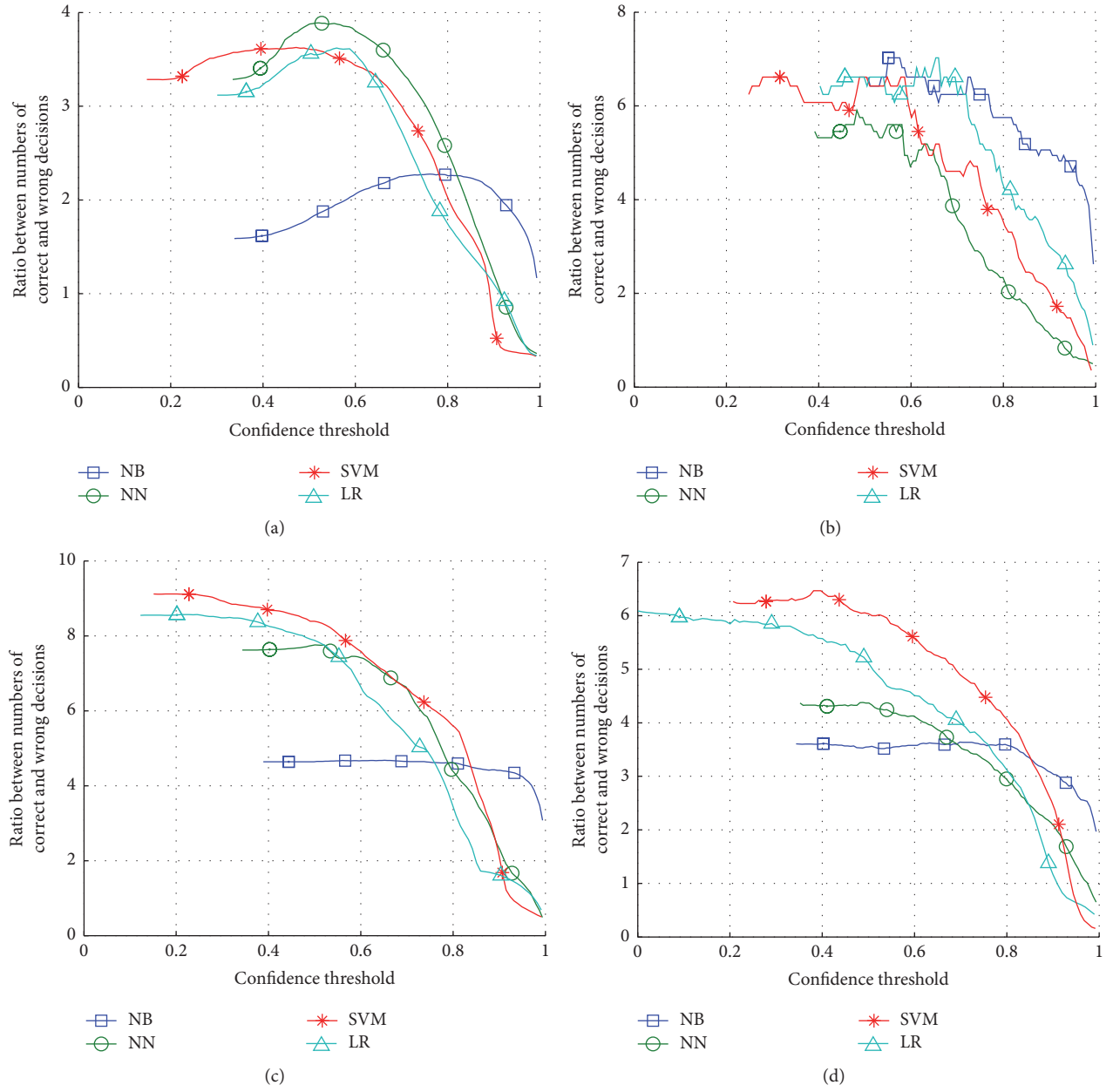



Figure 10: Ratio between the numbers of correct and wrong decisions as a function of confidence threshold. Data representations: data set #1, (a) visits, (b) places, and (c) cumulative samples; (d) data set #2, cumulative samples.

cumulative samples of the both data sets the threshold for LR rejects very few samples and the accuracy does not improve, as can be seen in Figures 11(c) and 11(d). With these data representations the improvements by the other classifiers are not significant either; with NB in Figure 11(d) the increase is about 0.03; in other cases it is about 0.01 or less. To summarize, with visits, the improvement obtained using confidence thresholds is more significant than with other data representations. However, even when applying thresholds, the accuracies are not as high as with places (compare the A bars of Figures 11(a) and 11(b)), but the difference is greatly reduced from Figures 3(a) and 3(b).

Comparing Figures 9 and 11, we see that the groups in Figure 9 with lower confidence and low accuracies, say below 0.5, have potential for accuracy improvements by rejecting results with low confidence, and the improvements are visible in Figure 11. However, based on these tests, with the data representations including averaging, the improvements are not significant.

In the results shown in Figures 9–11, also the determination of the thresholds is based on test data. Therefore, the effect of the threshold is not evaluated using independent data and, despite the modest improvements, these results may still be overly optimistic.

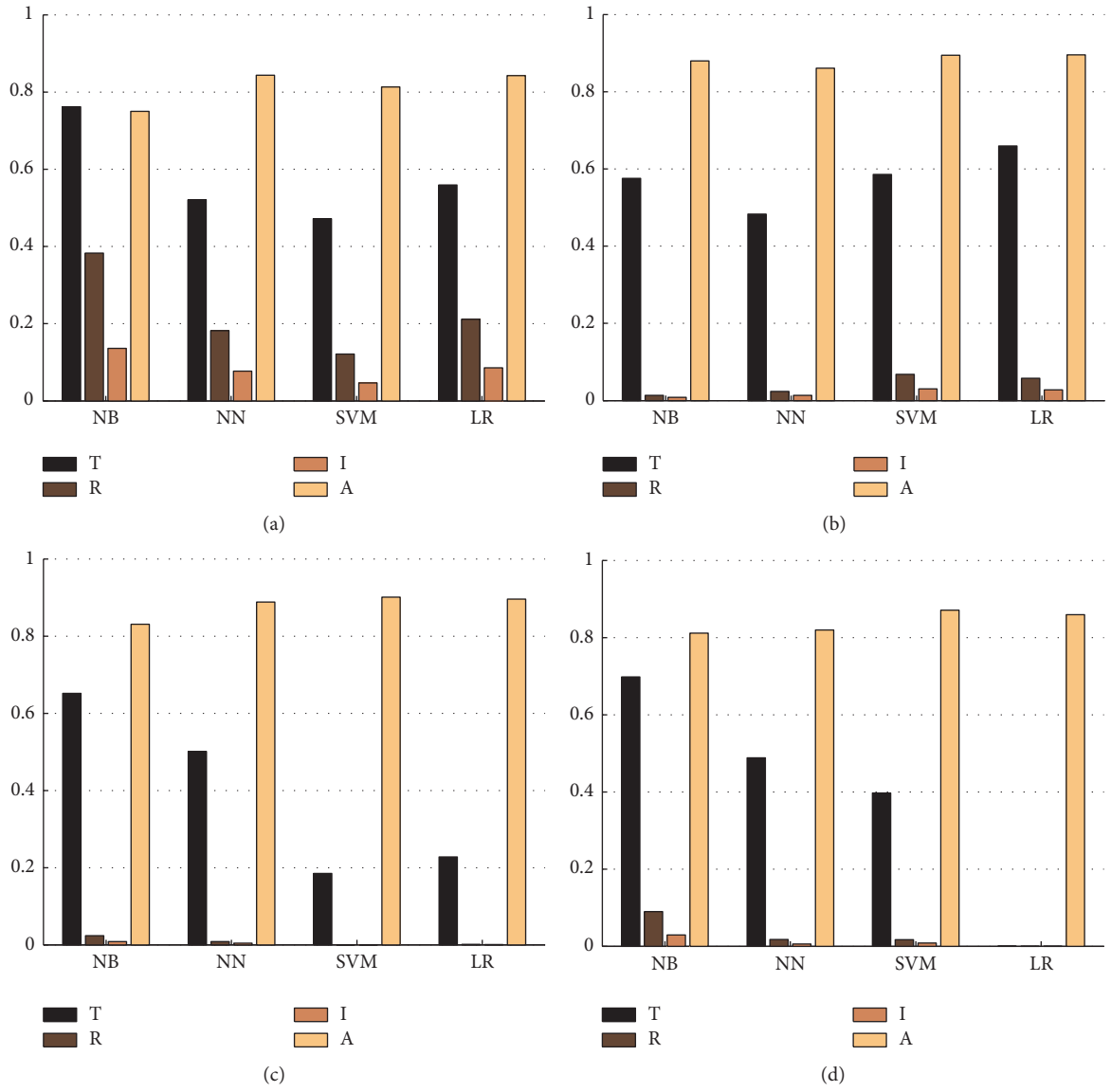



Figure 11: Rejecting samples with low classification confidence. T: confidence threshold, R: ratio of rejected samples to all samples, I: accuracy improvement obtained by rejecting low confidence samples, and A: classification accuracy within the samples with confidence ≥ threshold. Data representations: data set #1, (a) visits, (b) places, and (c) cumulative samples; (d) data set #2, cumulative samples. Accuracies given as ratios (unitless).

*5.5. Effect of Number of Classes.* In the previous tests we combined the less frequent places labels, such as friend's home, transportation, and restaurant into one class, *Other*. In this section, we compare these 3-class results to the 10-class classification results that we obtain with our classifiers and features. We used the same MDC data defined in Section 3.1 but now keeping the original 10 classes. We computed the places and cumulative samples representations from the 10-class data.

The comparison results are summarized in Table 3. We can notice that the numbers of cases are smaller in 3-class problem and the decrease comes from the decreased number of cases in classes other than *Home* or *Work*. We chose BT classifier for 10-class problem as it seems to outperform our other classifiers when number of classes is larger and compare it to LR of 3-class problem as LR performed well with both places and cumulative samples (Figure 3). For 10-class places representation, we computed two solutions, one using all the 14 features and another where we used forward SFS to select the most important features.

From the classification results it can be seen that adding more classes does not significantly affect the accuracy of *Home* and *Work*: the accuracies of *Home* are in both cases 92% or slightly better and the accuracies of *Work* are around 88%. However, the 10-class classifiers do not classify well the other places. With all features included, the overall accuracy is 62.3% and there are 4 classes that are never correctly classified. By reducing the number of features with SFS or

Table 3: Comparison of 3-class and 10-class solutions.

| Number of classes | 10 | | | 3 | |
|---|---|---|---|---|---|
| Data representation | Places | Places | Cum. Samples | Places | Cum. Samples |
| Number of cases | 369 | 369 | 128137 | 295 | 108531 |
| Home: number (percentage) | 106 (28.7) | 106 (28.7) | 39250 (30.6) | 106 (35.9) | 39250 (36.2) |
| Work: number (percentage) | 98 (26.6) | 98 (26.6) | 37602 (29.4) | 98 (33.2) | 37602 (34.6) |
| Other: number (percentage) | 165 (44.7) | 165 (44.7) | 51285 (40.0) | 91 (30.9) | 31679 (29.2) |
| Features | All 14 | 3: [4 1 2] | All 9 | All 14 | All 9 |
| Classifier | BT | BT | BT | LR | LR |
| Overall accuracy (%) | 62.3 | 68.5 | 68.4 | 89.2 | 89.5 |
| Class accuracies (%) | | | | | |
| 1 (Home) | 92.4 | 92.4 | 94.6 | 93.0 | 92.0 |
| 2 | 61.5 | 57.6 | 54.1 | | |
| 3 (Work) | 90.8 | 89.7 | 91.7 | 92.0 | 88.0 |
| 4 | 25.0 | 62.5 | 45.4 | | |
| 5 | 0.0 | 23.0 | 0.0 | | |
| 6 | 0.0 | 22.7 | 5.0 | | |
| 7 | 11.1 | 16.6 | 27.4 | | |
| 8 | 0.0 | 20.0 | 13.0 | | |
| 9 | 15.7 | 31.5 | 21.6 | | |
| 10 | 0.0 | 42.8 | 23.2 | | |
| (Other) | (26.0) | (40.0) | (31.2) | 81.0 | 89.0 |

Table 4: Comparison of data and solutions.

| Solution | Users | Percentage of cases | | | Labels | Features | Best classifier | Accuracy (%) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Home | Work | Other | | | | Overall | Home | Work |
| [6] | 80 | 25 | 30 | 45 | 10 | 2,769,200 | GBT | 75.1 | N/A | N/A |
| [7] | 80 | 25 | 30 | 45 | 10 | 54 | (1) | 65.8 | 87 | 85 |
| [8] | 80 | 25 | 30 | 45 | 10 | 1177 | (2) | 73.3 | 100 | 100 |
| [9] | 114 | 25 | 29 | 46 | 10 | 500 | (3) | 75.5 | 92 | 90 |
| #1 places | 114 | 29 | 26 | 45 | 10 | 3 (SFS) | BT | 68.5 | 92 | 90 |
| #1 cum. s. | 114 | 31 | 29 | 40 | 10 | 9 | BT | 68.4 | 94 | 92 |
| #1 visits | 114 | 52 | 38 | 10 | 3 | 14 | NN | 76.7 | 83 | 86 |
| #1 places | 114 | 36 | 33 | 31 | 3 | 14 | LR | 89.2 | 93 | 92 |
| #1 cum. s. | 114 | 36 | 35 | 29 | 3 | 9 | LR | 89.5 | 92 | 88 |
| #2 cum. s. | 16 | 31 | 26 | 42 | 3 | 11 | LR | 85.9 | 81 | 83 |

(1) Multilevel 2-method (SMO and simple logistic), fusion with decision tree. (2) Ensemble of binary classifiers using 1NN and SVM. (3) Combination of multiclass random forests and one-versus-all random forest binary classifiers.

using cumulative samples, the ability to classify also the less frequent places increases as shown in the bottom row, where average classification rates are computed for the other classes. Due to this improvement, the overall accuracy increases more than 6% to 68.5% and 68.4%. However, these are significantly lower than the overall accuracies of 3-class problem.

Based on this comparison, it is clear that with this type of user data, it is beneficial to combine the less frequent classes in order to classify better the more frequent and important places. Although the classification rates of *Home* and *Work* are on the same level in both 3-class and 10-class problems, the lower overall accuracies with 10-class indicate that there are more false detection of *Home* and *Work*.

## 6. Discussion

Papers [6–8] also aim at semantic place prediction and use data derived from the same database as data set #1 in our work. However, there are significant differences between their work and ours. Papers [6–8] are all from participants of the dedicated track on semantic place prediction in the Mobile Data Challenge (MDC) by Nokia, described in [5] and in more detail and with MDC outcomes in [36]. The data and findings based on it are described in [9], which also describes one solution of semantic place prediction. Basic information on the data, methods, and results of [6–9] and our work are summarized in Table 4.

The participants of the track used a subset of full MDC data that included the data of 80 users with the highest-quality location traces while we used the data of all the 114 users that had labeled visits data, without knowledge of quality of the data. The data used in [6–9] was from visits that lasted at least 10 minutes while our data was from visits that lasted at least 20 minutes. Therefore, their data included more cases from classes other than *Home* or *Work* compared to our data representations based on MDC data (data set #1). The difference is significant in visits representation but the accumulation of data changes these ratios. In data set #2, where the data collection has been implemented differently, the percentage of label *Other* is higher than the percentage of the other labels.

The numbers of extracted features are also given in Table 4. We used only 9–14 features related to time and phone usage but not to the environment while the other works used also environment related features such as number of Bluetooth or WLAN devices heard by the phone. We tested feature selection on data set #2 in both forward and backward directions but the results shown in the table were obtained using all 11 features. The authors in [7, 9] used feature selection method similar to our sequential feature selection in forward direction while in [6] they used two methods, Weka's Relief and L1-regularized logistic regression for the task.

The main focus in [6] is in generating a large number of conditioned features and then selecting the best features. The classification results using logistic regression, SVM with different kernels, Gradient Boosted Trees (GBT), and random forests are reported. The authors of [6] have published an extension to paper [21].

To give the final result, [7, 8] both use fusion of several classifiers or classification methods. Reference [7] uses multilevel classification model where labels are grouped so that in a sequence of classifications tasks with lower number of labels the algorithm selects label groups in hierarchical manner and finally in the lowest level chooses between two labels. In the paper, several methods are used to train different types of classifier models for multilevel classification. Then collection of these models is used to classify the data, and their classification results are used as a new feature vector that is used to train the final classifier.

Combination of smart binary classifiers is used in [8], where the multiclass classification problem is divided into a set of 2-class classification problems of types one-versus-one labels or one-versus-two labels. In the ensemble of binary classifiers each classifier uses the best combination of features for the current task and the better method from 1NN (i.e., KNN with $k = 1$) and SVM with RBF kernels. Three different methods for combining the classification outputs of the binary classifiers are evaluated in the paper.

In [9] three classification methods were used: (a) multiclass random forests, (b) one-versus-all random forest for each label where the winner class was decided by combining one-versus-all votes, and (c) combination of these. The accuracy of the methods was evaluated using leave-one-user-out cross validation similarly as in our comparisons.

In our work we solved 3-class problem with labels *Home*, *Work*, and *Other* instead of 10-class problem in [6–9]. We also used fewer features and simpler classifier models; that is, similarly as in [6] we did not use collections of classifiers except in BT (10 trees) and SVM (3 binary classifiers). The simpler models are generally preferred in resource constrained mobile devices. We also studied the effect of averaging of features by testing different data representations that include different levels of averaging: in visits representation each visit is classified separately, in cumulative samples, the features evolve with time as more data become available, and finally in places representation all data collected from one user in one place is averaged. For comparison, we also applied our features and classifiers to 10-class problem.

As we consider the memory consumption of SVM in classification phase too demanding for resource constrained mobile devices, we do not report its results in Table 4 even if it shows the best result with some data representations. In these cases, the results of the second best classifier are shown.

Due to the problem simplification from 10-class problem to 3-class in our approach and data retrieved from MDC database using slightly different criteria, the performance figures of Table 4 cannot be directly compared. However, due to the simpler task and despite the simpler classifier models, with visits representation and NN, we obtained the overall accuracy 76.7%, which is in the same level as the overall accuracy reported in the other works. With data representations including averaging the accuracies improve to 85.9% and better. The classification accuracies of *Home* and *Work* with places and cumulative samples of data set #1 are on the same level as in [7, 9]. With places representation where the data instances describe only short periods of time, these accuracies are lower as they are also with data set #2. In the latter case, the number of instances with label *Other* is higher than the numbers with the other labels, and, for this reason, the label *Other* is also classified with better accuracy (91%) than the two other labels.

The comparison between 3-class and 10-class problems with our classifiers and features show that our models can detect *Home* and *Work* reliably in both problems. The fact that in our model the inference is based on visits that are at least 20 minutes in duration may also contribute to this, as the shorter visits probably have phone usage characteristics that are closer to the decision borders. However, in 10-class problem the decreased classification rate of the less frequent places decreases the overall accuracy. Improving classification accuracy of the other places in 10-class problem requires using features that are directly related to environment, using phone usage data that is less privacy-preserving, and using more complex classifiers.

It can also be argued that MDC data is a bit old. As the MDC data set is from the time of the first smart phones, it does not describe well all the modern ways to use a smart phone. Through the evolution of new technologies, smart phone usage has changed a lot [37]. Nowadays, due to the internet connections available in phones, the use of SMS has decreased and messaging is often performed through other applications such as WhatsApp. The social media and messaging apps have reduced the need for voice calls and the

voice calls can also be made over internet based connections. Watching videos and TV on smart phones has become common as well as using social media and social games. With smart phones, photos are taken and videos recorded and both are shared in social media. Also the link between place and phone usage through the availability of WiFi networks is changing: the operators of wireless communication networks have started to bring inexpensive data plans with unlimited mobile data available to consumers, which allows them to use data-hungry applications also on the move [38].

## 7. Conclusion

We have developed an inference system to assign semantic place label for user's whereabouts based on the phone usage. The semantic places we considered in this work were Home, Work, and Other places. Our test results indicate that data representations that include averaging, that is, the places and cumulative samples representations, give higher classification accuracies than the visits representation. The average accuracy obtained using the visits representation is exceeded by the cumulative samples representation after only 2–6 days of accumulation of the data. Based on our preliminary tests with data set #2, the relevance of the features seem to depend on the classifier. However, features *stay*, *nightStay*, *headSet*, and *idle* seem to be relevant for most of the classifiers. Our tests also indicate that the classification accuracy can be improved by using thresholding based on classification confidence. The improvement was larger if the data representation did not include averaging.

*7.1. Future Work.* The future developments of the semantic labeling of user location context could include verification of the models using a bigger data set: more users, different life styles and daily patterns, different work occupations, and data for longer periods of time. The bigger data set could be used to learn subclasses to the current ones. In the group *Other* subclasses such as shop, restaurant, cinema, gym, outdoor exercising, lodging, leisure, and errands could be found. *Work* could include different kinds of work-like activities, such as shift work, driving work, other traveling work, attending school or university, and remote working from home. Also the use of *Home* is different for different people; for example, the elderly stay mainly at home.

In this study, we used bagged trees as an improved version of decision trees. Bagging improves variance of classifier by averaging/majority selection of outcome from multiple fully grown trees on variants of training set. Random forest is an interesting alternative for future work. It builds a collection of decorrelated trees by randomizing also the feature collection in the trees that are averaged (see, e.g., [31]).

## Disclosure

This work is an extension to our paper [20] in UPINLBS 2014.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This work was financially supported by Microsoft Corporation and by EU FP7 Marie Curie Initial Training Network MULTI-POS (Multi-Technology Positioning Professionals) under Grant no. 31652. The research in this paper used the MDC database made available by Idiap Research Institute, Switzerland, and owned by Nokia.

## References

[1] B. Heggestuen, "Smarthphone and tablet penetration," *Business Insider*, 2013, http://www.businessinsider.com/smartphone-and-tablet-penetration-2013-10.

[2] I. Lunden, "6.1b smartphone users globally by, 2020, overtaking basic fixed phone subscriptions," *TechCrunch*, 2015, https://techcrunch.com/2015/06/02/6-1b-smartphone-users-globally-by-2020-overtaking-basic-fixed-phone-subscriptions/#.t50cru:JF5k.

[3] B. Rao and L. Minakakis, "Evolution of mobile location-based services," *Communications of the ACM*, vol. 46, no. 12, pp. 61–65, 2003.

[4] N. Kiukkonen, J. Blom, O. Dousse, D. Gatica-Perez, and J. Laurila, "Towards rich mobile phone datasets: Lausanne data collection campaign," in *Proc. ACM Int. Conf. on Pervasive Services (ICPS)*, Berlin, Germany, 2010.

[5] J. K. Laurila, D. Gatica-Perez, I. Aad et al., "The mobile data challenge: Big data for mobile computing research," in *Proc. Mobile Data Challenge by Nokia Workshop, in Conjunction with International Conference on Pervasive Computing*, Newcastle, UK, June 2012.

[6] Y. Zhu, E. Zhong, Z. Lu, and Q. Yang, "Feature engineering for place category classification , in," in *Proceedings of the Proc. Mobile Data Challenge by Nokia Workshop*, Newcastle, UK, Newcastle, UK, June 2012.

[7] C.-M. Huang, J.-C. Ying, and V. S. Tseng, "Mining users behaviors and environments for semantic place prediction," in *Proceedings of the Proc. Mobile Data Challenge by Nokia Workshop*, Newcastle, UK, June 2012.

[8] R. Montoli, A. M. Us, J. M. Sotoca, R. Montoliú, and A. M. Usó, "Semantic place prediction by combining smart binary classifiers," in *Proceedings of the Proc. Mobile Data Challenge by Nokia Workshop*, Newcastle, UK, June 2012.

[9] T. M. T. Do and D. Gatica-Perez, "The places of our lives: Visiting patterns and automatic labeling from longitudinal smartphone data," *IEEE Transactions on Mobile Computing*, vol. 13, no. 3, pp. 638–648, 2014.

[10] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *Handheld and Ubiquitous Computing: First International Symposium, HUC '99 Karlsruhe, Germany, September 27–29, 1999 Proceedings*, vol. 1707 of *Lecture Notes in Computer Science*, pp. 304–307, Springer, Berlin, Germany, 1999.

[11] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 2, no. 4, pp. 263–277, 2007.

[12] O. A. Nykänen and A. Rivero Rodriguez, "Problems in context-aware semantic computing," *International Journal of Interactive Mobile Technologies*, vol. 8, no. 3, pp. 32–39, 2014.

[13] J. Kantola, M. Perttunen, T. Leppänen, J. Collin, and J. Riekki, "Context awareness for GPS-enabled phones," in *Proceedings of the Institute of Navigation - International Technical Meeting (ITM '10)*, pp. 287–294, 2010.

[14] L. Pei, R. Chen, J. Liu et al., "Motion recognition assisted indoor wireless navigation on a mobile phone," in *Proceedings of the 23rd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS '10)*, pp. 3366–3375, 2010.

[15] P. Zhou, Y. Zheng, Z. Li, M. Li, and G. Shen, "IODetector: A generic service for indoor outdoor detection," in *Proceedings of the 10th ACM Conference on Embedded Networked Sensor Systems (SenSys '12)*, pp. 113–126, 2012.

[16] A. Eronen, J. Leppänen, J. Collin, J. Parviainen, and J. Bojja, *Method and apparatus for determining environmental context utilizing features obtained by multiple radio receivers, patent Application US0053069*, 2013, http://www.google.com/patents/US20130053069.

[17] Android Developers, "Locale object," http://developer.android.com/reference/java/util/Locale.html.

[18] T. Do and D. Gatica-Perez, "By their apps you shall understand them," in *Proceedings of the the 9th International Conference*, pp. 1–10, Limassol, Cyprus, December 2010.

[19] A. Rahmati, C. Shepard, C. Tossell, L. Zhong, and P. Kortum, "Practical context awareness: measuring and utilizing the context dependency of mobile usage," *IEEE Transactions on Mobile Computing*, vol. 14, no. 9, pp. 1932–1946, 2015.

[20] A. Rivero-Rodriguez, H. Leppäkoski, and R. Piché, "Semantic labeling of places based on phone usage features using supervised learning," in *Proceedings of the Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS '14)*, pp. 97–102, 2014.

[21] Y. Zhu, E. Zhong, Z. Lu, and Q. Yang, "Feature engineering for semantic place prediction," *Pervasive and Mobile Computing*, vol. 9, no. 6, pp. 772–783, 2013.

[22] K. Farrahi and D. Gatica-Perez, "A probabilistic approach to mining mobile phone data sequences," *Personal and Ubiquitous Computing*, vol. 18, no. 1, pp. 223–238, 2014.

[23] T.-B. Nguyen, T. Nguyen, W. Luo, S. Venkatesh, and D. Phung, "Unsupervised inference of significant locations from WiFi data for understanding human dynamics," in *Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia (MUM '14)*, pp. 232–235, November 2014.

[24] E. S. Lohan and P. Figueiredo e Silva, "User traces analysis based on crowdsourced data," in *Proceedings of the 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1303–1308, Valencia, Spain, June 2017.

[25] E. Malmi, T. M. T. Do, and D. Gatica-Perez, "From foursquare to my square: Learning check-in behavior from multiple sources," in *Proceedings of the ICWSM*, Boston, MA, USA, 2013.

[26] T. M. T. Do, O. Dousse, M. Miettinen, and D. Gatica-Perez, "A probabilistic kernel method for human mobility prediction with smartphones," *Pervasive and Mobile Computing*, vol. 20, pp. 13–28, 2015.

[27] Microsoft, "Lumia sensorcore sdk 1.1 preview," https://msdn.microsoft.com/en-us/library/dn924551.aspx.

[28] "Machine Learning, Neural and Statistical Classification," D. Michie, D. J. Spiegelhalter, C. C. Taylor, and J. Campbell, Eds., Ellis Horwood, Upper Saddle River, NJ, USA, 1994.

[29] J. Han and M. Kamber, "Data Mining: Concepts and Techniques," Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, 2000.

[30] S. Russell and P. Norvig, *Artificial Intelligence, A Modern Approach*, Pearson Education Inc, 2003.

[31] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, New York, NY, USA, 2008.

[32] K. J. Cios, W. Pedrycz, and R. W. Swiniarski, *Data Mining Methods for Knowledge Discovery*, Springer US, Boston, MA, USA, 1998.

[33] S. Haykin, *Neural Networks and Learning Machines*, Perason Education, Inc, 2008.

[34] S. J. Delany, P. Cunningham, D. Doyle, and A. Zamolotskikh, "Generating Estimates of Classification Confidence for a Case-Based Spam Filter," in *Case-Based Reasoning Research and Development*, vol. 3620 of *Lecture Notes in Computer Science*, pp. 177–190, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[35] W. Cheetham, "Case-Based Reasoning with Confidence," in *Advances in Case-Based Reasoning*, vol. 1898 of *Lecture Notes in Computer Science*, pp. 15–25, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[36] J. K. Laurila, D. Gatica-Perez, I. Aad et al., "From big smartphone data to worldwide research: the Mobile Data Challenge," *Pervasive and Mobile Computing*, vol. 9, no. 6, pp. 752–771, 2013.

[37] Deloitte, *There's no place like phone*, Deloitte Global Mobile Consumer Survey, 2016, http://www.deloitte.co.uk/mobileuk/assets/pdf/Deloitte-Mobile-Consumer-2016-There-is-no-place-like-phone.pdf.

[38] Tefficient, "Unlimited pushes data usage to new heights," *Industry analysis*, 2016, http://tefficient.com/unlimited-pushes-data-usage-to-new-heights/.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 6268797, 19 pages
https://doi.org/10.1155/2017/6268797

*Research Article*

# An Efficient Normalized Rank Based SVM for Room Level Indoor WiFi Localization with Diverse Devices

**Yasmine Rezgui, Ling Pei, Xin Chen, Fei Wen, and Chen Han**

*Shanghai Key Laboratory of Navigation and Location-Based Services, School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China*

Correspondence should be addressed to Ling Pei; ling.pei@sjtu.edu.cn

Received 23 February 2017; Revised 3 May 2017; Accepted 11 May 2017; Published 9 July 2017

Academic Editor: Elena-Simona Lohan



This paper proposes an efficient and effective WiFi fingerprinting-based indoor localization algorithm, which uses the Received Signal Strength Indicator (RSSI) of WiFi signals. In practical harsh indoor environments, RSSI variation and hardware variance can significantly degrade the performance of fingerprinting-based localization methods. To address the problem of hardware variance and signal fluctuation in WiFi fingerprinting-based localization, we propose a novel normalized rank based Support Vector Machine classifier (NR-SVM). Moving from RSSI value based analysis to the normalized rank transformation based analysis, the principal features are prioritized and the dimensionalities of signature vectors are taken into account. The proposed method has been tested using sixteen different devices in a shopping mall with 88 shops. The experimental results demonstrate its robustness with no less than 98.75% correct estimation in 93.75% of the tested cases and 100% correct rate in 56.25% of cases. In the experiments, the new method shows better performance over the KNN, Naïve Bayes, Random Forest, and Neural Network algorithms. Furthermore, we have compared the proposed approach with three popular calibration-free transformation based methods, including difference method (DIFF), Signal Strength Difference (SSD), and the Hyperbolic Location Fingerprinting (HLF) based SVM. The results show that the NR-SVM outperforms these popular methods.

## 1. Introduction

With the fast growing of ubiquitous computing, the rapid advances in mobile devices, and the availability of wireless communication, wireless indoor positioning systems have become very popular in recent years. Moreover leveraging public infrastructure has many advantages such as cost efficiency, operational practicability, and pervasive availability.

Various short-range radio frequency technologies are broadly researched to address positioning task in GNSS denied area, for instance, Radio Frequency Identification (RFID) [1], Wireless Local Area Network (WLAN, a.k.a. WiFi) [2, 3], Bluetooth [4, 5], ZigBee [6], Ultra Wide Band (UWB) [7], and cellular networks [8]. All these means are high potential alternatives to indoor positioning. Meanwhile, indoor localization based on signals of opportunity (SoOP) is still a challenging task, since it requires stable interior wireless signals and adequate adaptation of the signals which are not originally designed for positioning purpose.

Fingerprinting method [9] is one of the most popular and promising indoor positioning mechanisms. It is a technique based upon existing WiFi infrastructure and thus requires no dedicated infrastructure to be installed. It allows positioning by making use of signal characteristics using the signature matching technique. Fingerprinting technique is accomplished by two steps. Firstly, it stores WiFi signatures from different radio wave transmitters for each reference position. Then, it compares the current signature of a device with prerecorded signatures to find the closest match.

Different techniques and solutions have been proposed providing a varying mix of resolution, accuracy, stability, and challenges. Early examples of a positioning system that uses fingerprinting are RADAR [2] and Horus [10] systems. Horus system is based on the probabilistic approach which considers the statistical characteristics of the RSSIs and their distribution. The Lognormal distribution [10], Weibull function [5], Gaussian distribution [11], and the double peak

Gaussian distribution [12] were used to model the RSSI distribution. However, RADAR system was the first introducing the fingerprinting technique based on the deterministic approach [13] using $K$-Nearest Neighbor method (KNN) [14]. Nowadays, the use of machine learning algorithms (ML) as in [15] has increasingly gained more popularity in indoor navigation domain because of their witnessed robustness; among them are Naïve Bayes classifier [16, 17], Support Vector Machine (SVM) [17, 18], Random Forest [18, 19], and Neural Network [20, 21]. However, their model generalization to different user's terminals has seldom been considered.

RSSI fluctuation and diversiform smartphones can significantly degrade the positioning accuracy of WiFi localization systems, as well as the patterns between training and testing signature vectors. The WiFi device used to train the radio map during the calibration phase may especially differ from the ones used during the positioning phase. WiFi modules from different providers have varying receive signal gains which make the RSSI vary using different devices at the same location [22]. This hardware variance problem is not only limited to differences in the WiFi chipsets used by training and tracking devices. Besides, it arises when the same WiFi chipsets are connected to different antenna types and/or packaged in different encapsulation materials.

To address this issue, several studies have proposed methods to improve the robustness of positioning systems against mobile devices heterogeneity. For example, the use of an unsupervised learning [23], the Hyperbolic Location Fingerprinting (HLF) [24], the DIFF method [25], and the Signal Strength Difference method (SSD) [26] are the representative ones. In this paper, a new method applying an intermediate step of absolute RSSI value transformation by making use of ML algorithms is proposed.

The indoor navigation market addresses various applications like Location Based Services (LBS), the guidance of firefighters, and people management. Moreover, it can benefit from the room level accuracy [27–29] to extract statistical information which can be deployed to better market to customers, find hangouts in airports, and even most popular shop in shopping malls, and so forth. In this work, we focus on a room localization which is the prediction of an occupied room in which the mobile device is currently in. This task is more practical to attain since it does not require the floor map of the desired indoor environment which is not always available in practice.

To achieve a high room level accuracy based WiFi fingerprinting technique, we have investigated various database definitions and the impact of signature clustering in our previous work [30], with respect to relevant requirement parameters. In this paper, we propose a normalized rank transformation based SVM approach to solve the issue of mobile terminal diversity during the positioning phase. We consider and compare between the performances of the aforementioned ML algorithms and calibration-free transformations to validate our solution. This approach aims at generalizing the use of the preconstructed radio map derived from one device to manifold devices and guarantee localization accuracy in the meantime.

## 2. Normalized Rank Transformation

Aiming to mitigate the effect of signal fluctuation and hardware variance issue, we introduce the intermediate normalized rank transformation step to freeze the variation of the RSSI. This solution has been defined principally to deal with SVM classifier. Moving from RSSI value based analysis to the normalized rank transformation based analysis, the principal features are prioritized and the dimensionalities of signature vectors are considered.

*2.1. Rank Transformation.* In the conventional classification based on SVM or any other ML algorithms for fingerprinting-based indoor localization, the features are defined as the signal strength received from all the visible access points (APs) to build the model. However, the received signal strength is inherently time varying at a specific location. Moreover, device diversity will impact both learning and positioning phases.

In order to find the perfect match between the user location and the predefined locations in the radio map using SVM classifier, as well as attenuate the susceptibility of decision boundaries to RSSI variation, the enhancement of data learning to build a robust model is a key. The improvement of the accuracy is related to the strong decision boundaries between the different classes. The input variables are redefined and the absolute RSSI values are replaced first by their corresponding rank values. Let $P_{1:M}$ be the RSSI vector of the $M$ visible APs and $S_i$ the corresponding absolute RSSI value of the AP($i$) such that

$$P_{1:M} = \{S_1, S_2, \ldots, S_M\} \tag{1}$$

$$P'_{1:M} = \{S'_1, S'_2, \ldots, S'_M\} \tag{2}$$

$$S'_1 \leq S'_2 \leq \cdots \leq S'_M \tag{3}$$

$$\Psi_i = \begin{cases} \Psi_{i-1} + 1 & \text{for } S'_{i-1} \neq S'_i \\ \Psi_{i-1} & \text{otherwise.} \end{cases} \tag{4}$$

$P'_{1:M}$ is the new defined RSSI vector by rearranging the APs. The rearrangement is based on (3), in which the throughputs of received signals are in ascending order. Equation (4) explains how to decide the rank vector. If RSSIs values are distinct ($S'_{i-1} \neq S'_i$, $i$ indicates the position of the AP in the $M$ dimensional vector), successive numbers denoted by $\Psi_i$ will be assigned. Otherwise, the same rank will be allocated. The initial value is equal to one and $\Psi_M$ does not have to be equal to the dimension of the initial RSSI vector. By adopting this procedure, the most reliable APs with high RSSI are tagged with the high rank values.

*2.2. Normalized Rank Transformation.* The observed set of APs is not fixed over time at every calibration point and, consequently, the dimension of the transformed rank vectors.

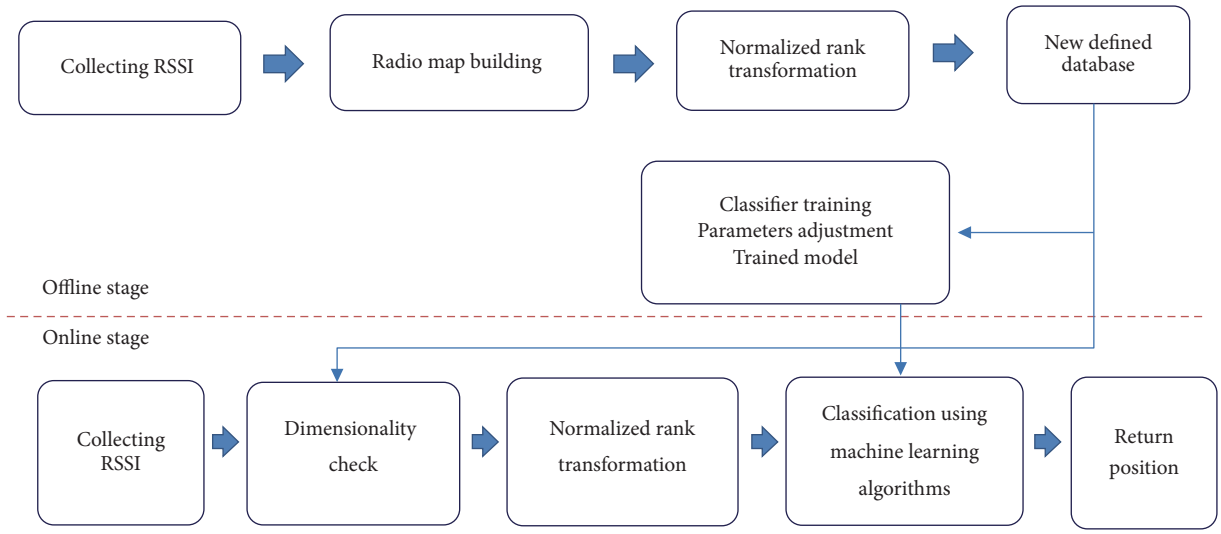



Figure 1: Proposed system architecture.

Hence, the new formulated rank vector has to be adapted to the varying assigned tags and needs to be normalized.

$$\mathrm{NR} = \{\lambda\Psi_1, \lambda\Psi_2, \ldots, \lambda\Psi_M\} \quad \text{such that } \lambda = \frac{1}{\Psi_M} \tag{5}$$

$$\mathrm{NR}_i \in [0, 1] \quad \text{such that } \mathrm{NR}_i = \lambda\Psi_i. \tag{6}$$

Equation (5) defines the normalized rank vector which is denoted as NR and satisfies (6), where $\lambda$ is the normalization factor corresponding to the inverse of the highest assigned rank $\Psi_M$ in the transformed rank vector. This normalization step will result in keeping the prioritization of the most reliable APs. Furthermore, it helps attenuating the effect of the noisy points as the hyperplanes are always dominated by the high values.

## 3. Proposed Method

In this section, we give more details about the proposed approach which is composed of two main stages: the offline stage and the online stage, as illustrated in Figure 1.

*3.1. The Offline Stage.* This stage consists of 3 major steps: *Step 1:* data collection and association; *Step 2:* normalized rank transformation; *Step 3:* model building and parameters adjustment.

*Step 1* (data collection and association). Selection of the sampling Reference Point within the region of interest is required to further collect the RSSI by a mobile node from all the available APs. Since the propagation of the radio signal in indoor environments is very complicated, several observations in the same location are needful. A single signature for each location is assigned by combining (such as through averaging) $x$ sampling times. Let $R$ be the total number of studied rooms and $M$ the total number of APs. $(R+1)\times(M+1)$ is the radio map matrix where each row vector is along with its corresponding position except the first $M$ dimensional vector space, which contains the MAC addresses of the $M$ visible access points denoted by $(\mathrm{AP})_{RM}$.

*Step 2* (normalized rank transformation). This intermediate step is achieved by separately considering each row vector of the radio map. This consideration follows the described steps in Section 2 proceeding from (1) to (6).

*Step 3* (model building and parameters adjustment). The model is built based on the new defined database and the transformed normalized rank values. The parameters of the learning method are tuned upon the obtained performance using the same device that is utilized to construct the radio map. The scrutinized criteria are the accuracy, precision, and the recall such that

$$\begin{aligned} \text{Accuracy} &= \frac{\mathrm{TP} + \mathrm{FN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \\ \text{Precision} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \\ \text{Recall} &= \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \end{aligned} \tag{7}$$

where TP, FP, TN, and FN are True Positive, False Positive, True Negative, and False Negative, respectively. Classification accuracy alone may hide details about the performance of classification model. It can also be misleading in the case of having more than two classes in the dataset. It gives the overall performance of the model considering all correct predictions divided by the total number of the datasets. However, it does not point out if all classes are predicted equally or whether some classes are neglected by the model. The formulations of the precision and the recall are used in our study to extract more information from the generated model. The precision gives the percentage of the correctly predicted instances among the total number of positive predictions, while the recall is the percentage of correctly predicted instances among the total number of positives.

(I) Dimensionality check
(1) **for** $j = 1, 2, \ldots, M$ **do**
(2) **for** $i = 1, 2, \ldots, N$ **do**
(3) **if** $[(\mathrm{AP})_{RM}]^{(j)} = [(\mathrm{AP})_{\mathrm{Ob}}]^{(i)}$
(4) Keep $(\mathrm{RSSI})^{(i)}$ of Ob vector at the position $(j)$
(5) **elseif** $i = N$ & $[(\mathrm{AP})_{RM}]^{(j)} \neq [(\mathrm{AP})_{\mathrm{Ob}}]^{(i)}$
(6) Pad position $(j)$ with zero value
(7) **Loop**
(II) Normalized rank transformation
(i) Keep only non-zero positions and get the new $\mathrm{Ob}'$ vector
(ii) Rearrange RSSI value in ascending order.
(iii) Assign transformed Normalized Rank value:
(1) Initialize the Normalized Rank (NR-Ob) vector
(2) **for** $k = 2, 3, \ldots,$ iterator **do**
(3) **if** $(\mathrm{Ob}')_k = (\mathrm{Ob}')_{k-1}$
(4) $(\mathrm{NR\text{-}Ob})_k = \Psi_{k-1}$
(5) **else**
(6) $(\mathrm{NR\text{-}Ob})_k = \Psi_{k-1} + 1$
(7) **Loop**
(8) $\lambda = \max\{(\mathrm{NR\text{-}Ob})_{(1:\mathrm{iterator})}\}$
(9) Normalize NR-Ob vector by $\lambda$ and remapping to $M$ dimensional space
(10) Consider the transformed normalized rank vector for positioning
(III) ML Classification
(IV) Return the final position

Algorithm 1: Proposed normalized rank transformation method during runtime phase.

*3.2. The Online Stage.* This part is fully detailed in Algorithm 1. Like the offline phase, this stage consists of three main steps: *Step 1:* dimensionality check; *Step 2:* normalized rank transformation; *Step 3:* ML classification and position estimation.

*Step 1* (dimensionality check). During our database construction, considering $M$ visible APs leads to the need of dimensionality check. This step checks each given observed vector at the location, which is denoted (Ob) and is to be identified. Let $N$ be the total number of visible APs of (Ob) vector. Dimensionality transformation from $N$ dimension space to $M$ dimension space is established such that the RSSI value at $i$ location in (Ob) vector corresponding to a specific AP at $j$ location in $(\mathrm{AP})_{RM}$ vector should be stored at the same $j$ location. However, for the unseen APs, their corresponding RSSI values are padded with zero to achieve the same dimension in both vectors. Furthermore, it allows pattern matching to take the same features into consideration during the location estimation process.

*Step 2* (normalized rank transformation). The RSSI values are transformed to the normalized rank values by keeping the $M$ dimensional space of the observed vector.

*Step 3* (ML classification and position estimation). The transformed normalized rank vector (NR-Ob) is compared with the trained model to find the best match. The physical position of the model which has the best match in the new radio map will be labeled as the estimated position.

The same RSSI transformation has been applied in both offline and online stages. Several classifiers will be trained using the new radio map, generating the training model to validate our approach.

## 4. Classification Methods

In this section, we introduce the different methods adopted in our work. All the studied algorithms fall under the category of the supervised techniques.

*4.1. Support Vector Machine.* Support Vector Machine is one of the best off-the-shelf supervised learning algorithms, which is used for both classification and regression tasks. It has been originally developed for binary classification problems and further expanded to perform even multiclass classification tasks. It uses hyperplanes to define decision boundaries separating data points of different classes in the case of linearly separable data. In addition, SVMs can efficiently perform a nonlinear classification by using kernel trick. In this case, original data are mapped into a high-dimensional, or even infinite-dimensional, feature space [31, 32].

SVM aims to construct a hyperplane with the maximal margin between different classes. In most cases, data are not perfectly linearly separable, which makes the separating hyperplane susceptible to outliers. Therefore, a restricted number of misclassifications should be tolerated around the margins. The resulting optimization problem for SVMs,

where a violation of the constraints is penalized, depends on the regularization norm considered.

*4.1.1. L1 Regularization Norm for SVM.* The L1 norm defines a Support Vector Machine with a linear sum of the slack variable to make the hyperplanes less sensitive to outliers; it is written as

$$\min_{w,\xi,b} \quad J(w,\xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \tag{8}$$

$$\text{Subject to:} \quad y_i\left(w^T\phi(x_i)+b\right) \geq 1-\xi_i \quad i=1,\ldots,N,\ \xi_i \geq 0,\ i=1,\ldots,N \tag{9}$$

$$y_i = \operatorname{sign}\left(w^T\phi(x_i)+b\right), \tag{10}$$

where $J$ is the cost function, $w$ is the weight vector, and $C$ is the positive regularization constant which defines the tradeoff between complexity and proportion of nonseparable samples. The problem formulation in (8) and (9) refers to the primal optimization problem. Introducing the Lagrange multipliers $\alpha_i \geq 0$, we obtain the following dual problem:

$$\max_{\alpha} \quad W(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j \phi(x_i)^T \phi(x_j) \tag{11}$$

$$\text{Subject to:} \quad \sum_{i=1}^{N} y_i\alpha_i = 0, \quad 0 \leq \alpha_i \leq C.$$

The optimal hyperplane is the one which maximizes the margin, and the optimal values of $w$ and $b$ are found by solving a constrained minimization problem using Lagrange multipliers. The function that solves the quadratic programming problem, and with the use of the positive definite mapping function that satisfies Mercer's condition, is such that

$$k(x,x_i) = \phi(x)^T\phi(x_i). \tag{12}$$

It also satisfies the Karush-Kuhn-Tucker (KKT) conditions.

The decision function can be expressed as

$$y_i = \operatorname{sign}\left(\sum_{i=1}^{N}\alpha_i y_i k(x,x_i) + b\right). \tag{13}$$

The classification accuracy produced by SVMs may show variations depending on the choice of the kernel function and its parameters. Various types of kernels can be chosen:

(i) Linear:

$$K(x,x_i) = x^T x_i. \tag{14}$$

(ii) Polynomial of degree $d$:

$$K(x,x_i) = \left(\gamma + x^T x_i\right)^d, \quad \gamma \geq 0. \tag{15}$$

(iii) Radial basis function (RBF):

$$K(x,x_i) = \exp\left(-\frac{\|x-x_i\|_2^2}{\delta^2}\right). \tag{16}$$

(iv) Sigmoid:

$$K(x,x_i) = \tanh\left(k_1 \cdot x^T x_i + k_2\right)^d. \tag{17}$$

The KKT condition is given by

$$\alpha_i\left(y_i\left(w^T x_i + b\right) - 1 + \xi_i\right) = 0 \qquad b_i\xi_i = (C-\alpha_i)\xi_i = 0. \tag{18}$$

$\alpha_i$ are Lagrange multipliers and the value $\xi_i$ indicates the distance of $x_i$ with respect to the decision boundary since that

(i) if $\alpha_i = 0$, then $\xi_i = 0$; therefore $x_i$ is correctly classified and lies outside the margin;

(ii) $0 < \alpha_i < C$; then $(y_i(w^T x_i + b) - 1 + \xi_i) = 0$ and $\xi_i = 0$; thus $y_i(w^T x_i + b) = 1$ and $x_i$ is the support vector;

(iii) $\alpha_i = C$; then $(y_i(w^T x_i + b) - 1 + \xi_i) = 0$ and $\xi_i \geq 0$; therefore $x_i$ is a bounded support vector; in the case $0 \leq \xi_i < 1$, $x_i$ is correctly classified; however, for $\xi_i \geq 1$, $x_i$ is misclassified.

*4.1.2. L2 Regularization Norm for SVM.* The L2 norm defines a Support Vector Machine which uses the square sum of the slack variable in the objective function. The considered optimization problem is as follows:

$$\min_{w,\xi,b} \quad J(w,\xi) = \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_{i=1}^{N}\xi_i^2$$

$$\text{Subject to:} \quad y_i\left(w^T x_i + b\right) \geq 1-\xi_i \quad i=1,\ldots,N,\ \xi_i \geq 0,\ i=1,\ldots,N. \tag{19}$$

The dual problem is expressed by introducing the Lagrange multipliers $\alpha_i$:

$$\max_{\alpha} \quad W(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j \left(k(x,x_i) + \frac{\delta_{ij}}{C}\right) \tag{20}$$

$$\text{Subject to:} \quad \sum_{i=1}^{N} y_i\alpha_i = 0, \quad \alpha_i \geq 0 \text{ for } i=1,\ldots,N,$$

where $\delta_{ij}$ is Kronecker's delta function, in which $\delta_{ij} = 1$, for $i = j$, and 0 otherwise.

The KKT conditions in this case are given by

$$y_i\left(\sum_{j=1}^{N}\alpha_j y_j\left(k(x,x_i) + \frac{\delta_{ij}}{c}\right) + b\right) - 1 = 0. \tag{21}$$

*4.2. KNN.* $K$-Nearest Neighbor algorithm is a nonparametric supervised classifier in which the target label is predicted by finding the nearest neighbor class, considering the majority vote of its $K$ neighbors. In the case of $K = 1$, the target is simply assigned to the class of its nearest neighbor. The closest class in this work is identified using the Euclidean distance:

$$D_{ij} = \sqrt{\sum_{k=1}^{M} \left(X_{ik} - X_{jk}\right)^2}, \tag{22}$$

where $D_{ij}$ is the distance between the observed point and each signature vector in the radio map. $X_i$ corresponds to the runtime fingerprint, $X_j$ is the offline fingerprint, and $M$ corresponds to the dimension of the RSSI Vector. The best choice of the calibration points $K$ selected in this work has been set equal to one, since it generated better results.

*4.3. Naïve Bayes.* Naïve Bayes is a probabilistic classifier based on applying Bayes theorem with strong naïve independence assumptions between the features. Each data instance generates a tuple $X$ of attribute values $\langle x_1, x_2, \ldots, x_M \rangle$. The identification of the corresponding label from a finite set of labels $R$ consists on maximum a posteriori (MAP) estimation. The theorem states the following relationship:

$$P\left(r \mid x_1, x_2, \ldots, x_M\right) = \frac{p\left(r\right) P\left(x_1, x_2, \ldots, x_M \mid r\right)}{P\left(x_1, x_2, \ldots, x_M\right)}, \tag{23}$$

since $P(x_1, x_2, \ldots, x_M)$ is constant given the input set:

$$\begin{aligned} r &= \arg\max p\left(r\right) P\left(x_1, x_2, \ldots, x_M \mid r\right) \\ r &= \arg\max p\left(r\right) \prod_{i=1}^{M} P\left(x_i \mid r\right), \end{aligned} \tag{24}$$

where $r$ is the estimated room, which is predicted given the transformed rank values from $M$ APs, $p(r)$ is the prior probability of the class "$r$," and $P(x_i \mid r)$ corresponds to the likelihood. Both terms are estimated from the training data. We adopt the implementation of Naïve Bayes considering the fact that continuous variable with each class is distributed according to a Gaussian distribution:

$$P\left(x_i = v \mid r\right) = \frac{1}{\sqrt{2\pi\delta_r^2}} e^{-(v-u_r)^2/2\delta_c^2}. \tag{25}$$

*4.4. Random Forest.* Random Forest (RF) is a classifier in which a multitude of decision trees are generated. The RF chooses the tree which has the highest votes after their classification results. The most occurring class number in the output of the decision trees is the final output of the RF classifier. A recursive process in which the input dataset is composed of smaller subsets allows the training of each decision tree. This process continues until all the tree nodes reach the similar output targets. The Random Forest classifier takes weights based on the input as a parameter that resembles the number of the decision trees [18].

*4.5. Artificial Neural Network.* Artificial Neural Network (ANN) is one of the most effective models in ML. It has been inspired by the biological neural networks in the human brain. It is made of several units or neurons of the following form:

$$H_j = \sigma\left(b + \sum_{i=1}^{N} w_{ij} x_i\right), \tag{26}$$

where $\sigma$ is a nonlinear activation function, $b$ is the bias term, and $w_{ij}$ are the associated weights to the column vector $x$ (corresponding either to the input data or the preceding layer).

In this paper, our selected activation function is the sigmoid function as in the following [20]:

$$\sigma\left(x\right) = \frac{1}{1 + e^{-x}}. \tag{27}$$

The nodes in ANN are structured into successive layers: input layer which corresponds to the input data, hidden layers, and output layer. The required number of hidden layers depends on the nonlinearity of the relation between input and output. Backpropagation algorithm is used to adjust weights and bias values of the edges and to minimize the loss function.

## 5. Experimental Results and Analysis

Our experiments were conducted in Metro City shopping mall in Shanghai and we considered about 88 different shops during the calibration phase. Figure 2 shows one floor plan of this shopping mall. Only one device has been used to build the radio map based on the collected RSSI measurements. The total visible $M$ APs considered herein are equal to 185. Samsung Galaxy Note 2 mobile phone is considered as the reference device.

During the testing phase, sixteen different devices have been utilized to investigate hardware variance effect on localization accuracy. These devices recorded the signal strength from the available APs with their corresponding MAC addresses at the same locations in 8 shops. In each shop, sixty samples have been recorded, with a total of 480 samples based on each single device. Different database sizes denoted by $S$ (corresponding to the number of assigned observations to each position) have been utilized to investigate how much we need to know ahead of time about what is being learned. This analysis aims to achieve an effective learning and correctly predict the position, for new observations unseen before. Moreover, the accuracy based on a single observation, as well as the needed fused data, is studied. The performance of the proposed method is evaluated through extensive experiments, and the obtained results based on SVM are compared with other well-known and widely used ML algorithms in indoor localization. In this section, we assess the performance of the built model based on a reference device to diverse devices. In the first part, we investigate database definition based on the collected RSSI value. It is implemented by assigning a varied set of observations to each single location and maintaining the absolute RSSI value during the runtime

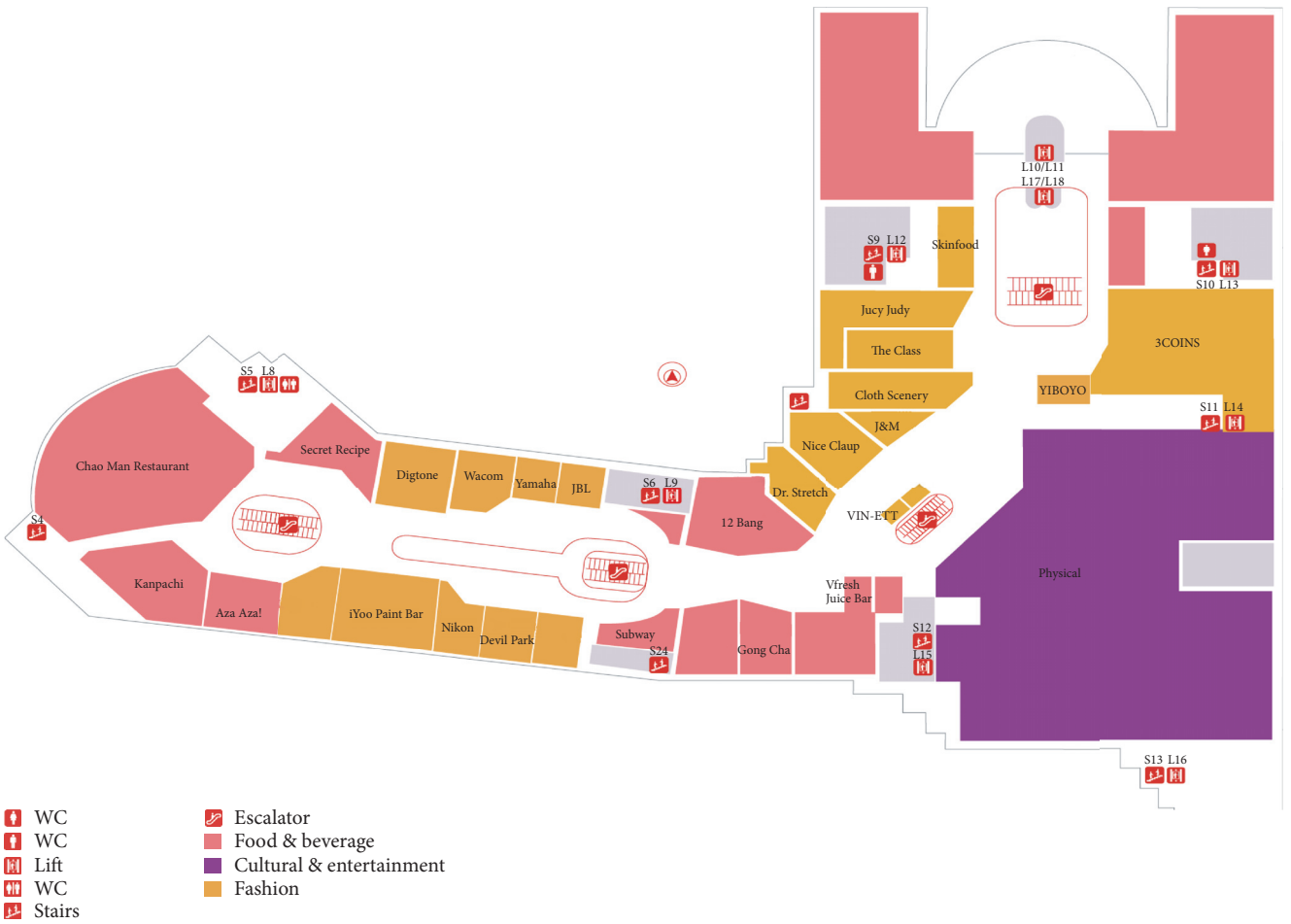



Figure 2: 3rd floor plan of Metro City shopping mall.

phase. The predicted location is calculated by matching the sample points on the radio map, with the RSSI fingerprint closest to the tracking device.

The linear kernel of SVM based on risk minimization principle is adopted to make the decision about the current location. Seeing that, for the most part of analysis, the number of input variables exceeds the number of examples, which makes the linear separation well suited based on Cover's theorem [33], same kernel parameters are considered during the whole analysis to fairly compare the obtained results. Moreover, we checked the efficiency of the rank transformation without any normalization. In addition, to evaluate the effectiveness of the proposed method, we considered about two ways of normalization. The first technique is the fully normalized rank, considering the total stored input in the database. However, the second one is the vector normalized rank method in which the normalization factor $\lambda$ is taking the value as described in Section 2.

*5.1. Experiments Based on RSSI Values.* Figure 3 shows the positioning accuracy with different $S$ and the influence of changing the tracking device when using RSSI values in both online and offline phase. The vertical axis shows the percentage of correctly predicted positions. We look first at the achieved accuracy in the case of using the same training device in the positioning phase. The good and steady performance during this test can be seen with high precision to estimate the location, where only a few samples are enough to attain the maximum accuracy. Almost all locations are perfectly estimated in this case.

However, in the case of tracking device different from the training device during the runtime phase, it works badly in half of the studied cases. Besides, this observation is much more remarkable if the number of fingerprint observations in the same location is limited or very small. Although the achieved correct rate for the remaining smartphones turns around 100% well-estimated location, only some of them can be distinguished in the figure. Unsteady curves are noticeable and increasing the number of the encoded fingerprints based on RSSI values at each location on the radio map does not always improve the performance of the applied method. This outcome proves the degradation pattern caused by the change of the utilized device to record the signal strength. This is due to the fact that RSSI stored in the database diverges from the RSSI captured using another piece of equipment. This low estimation is coming from the different sensitivity of

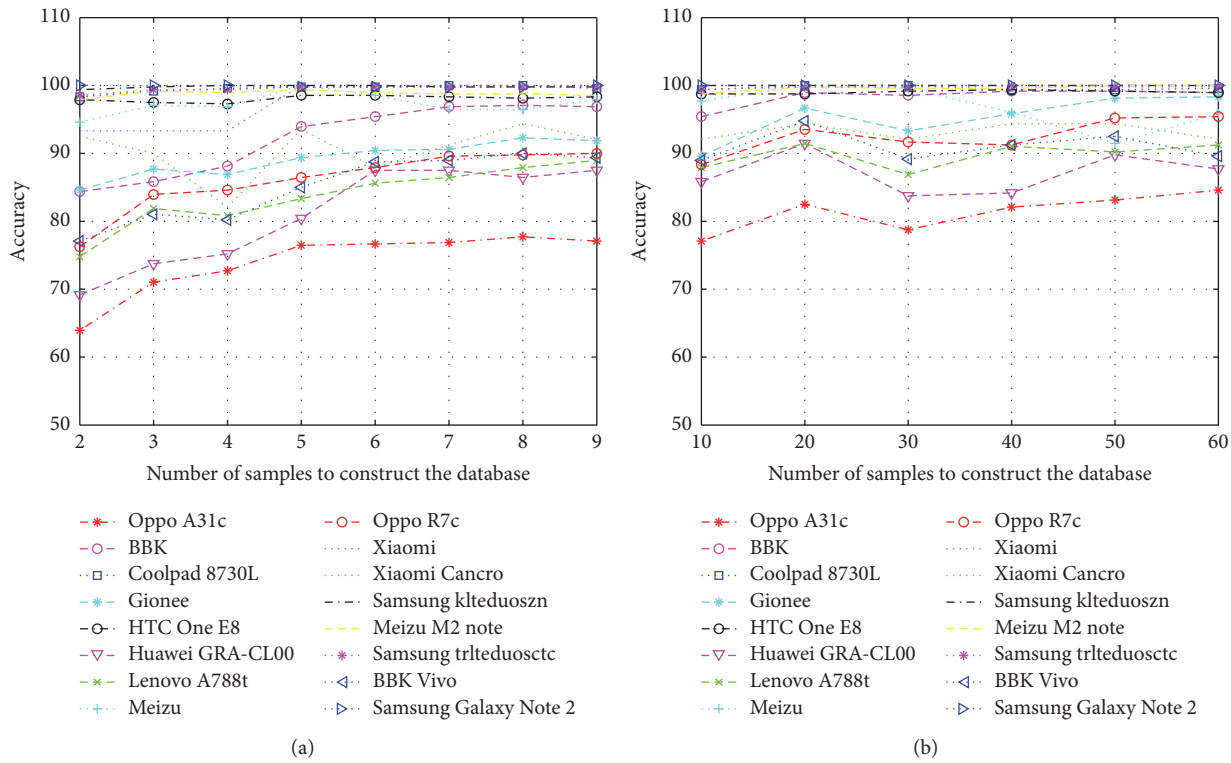



Figure 3: Localization accuracy using RSSI based on linear SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

manifold devices to the signal throughput, which is related to both antennas and packaging materials.

*5.2. Experiments Based on the Rank Transformed Values.* In this part, we have redone the previous tests based on the rank transformed values instead of exploiting the direct RSSI value to predict the location. The RSSI vector $P_{1:M} = \{S_1, S_2, \ldots, S_M\}$ is reformulated such that $\Psi_{1:M} = \{\Psi_1, \Psi_2, \ldots, \Psi_M\}$, where $\Psi_i$ satisfies (4) and the normalization factor is initially ignored. Figure 4 illustrates the location accuracy by changing the number of associated observations to each Reference Point (RP) using multiple devices. A prominent improvement is noticeable using the rank transformed value and more stability is perceptible comparing to the previous results. The maximum correct rate estimation is much higher based on a few associated samples to the reference locations. The reached perfect predictions based on this test have not been attained based on RSSI analysis even when we considered the integrality of captured observations. The two exception cases of *Xiaomi* and *Oppo A31c* devices will be discussed in the next part.

Ranking the signal throughput of an access point at a specific location plays an essential role in delimiting the range of the interval in which the input variable is defined. It is additionally practical to broaden the application of a built model. Moreover, the prioritization of the most reliable input variables when assigning rank values strengthens the generalization of the model and the performance of the used method.

*5.3. Experiments Based on the Normalized Rank Transformation Method.* In order to evaluate the performance of the normalized rank transformation, we define two means of normalization. Figure 5 represents the performance of the learning method using the fully normalized rank transformation. The normalization factor $\lambda$ takes the inverse value of $M$ where $M$ is the total visible APs considered in the radio map, while Figure 6 corresponds to the achieved results when $\lambda$ satisfies the condition of (5). It is noteworthy that both ways can provide more accurate results with some meaningful differences.

It appears that normalizing based on a fixed value needs more samples to attain a good prediction if compared to the second normalization side by side. It can be seen that, for the greatest part, at least 6 samples are required to reach an approximate 100% accurate prediction. The recorded RSSI values using *Xiaomi* device have been compared with those obtained with the remaining devices, where an unpredicted variability has been registered. We notice that some samples are very similar for the same environment. Meanwhile, it could exhibit a remarkable inconsistency in the recorded throughput at the same position. It seems that this mobile phone is very sensitive to changes and to the stability of the studied area which explains the registered uncertainty. The maximum percentage of 87.5% achieved by *Oppo A31c* is further interpreted based on the confusion matrix.

Nonetheless, the second followed normalization based on the highest assigned transformed rank value is rapidly converging to the maximum accuracy, except for the special

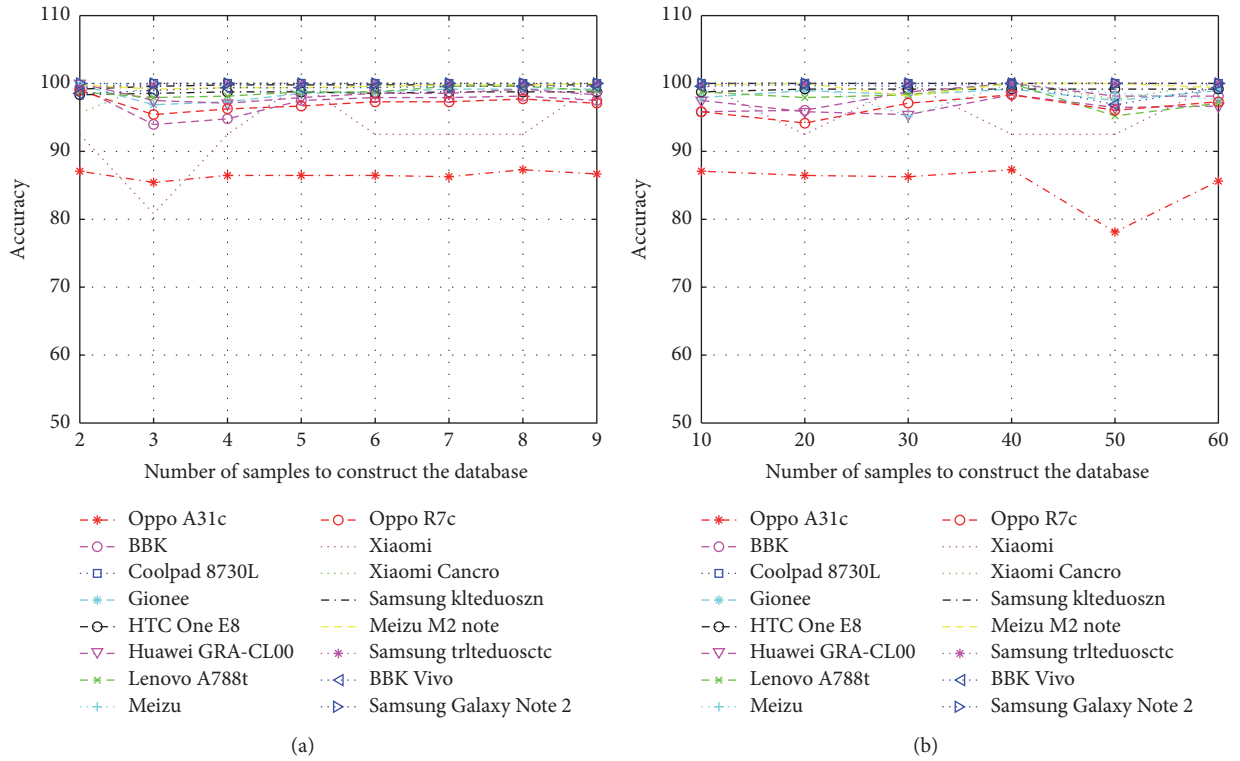



Figure 4: Localization accuracy using the rank based on linear SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

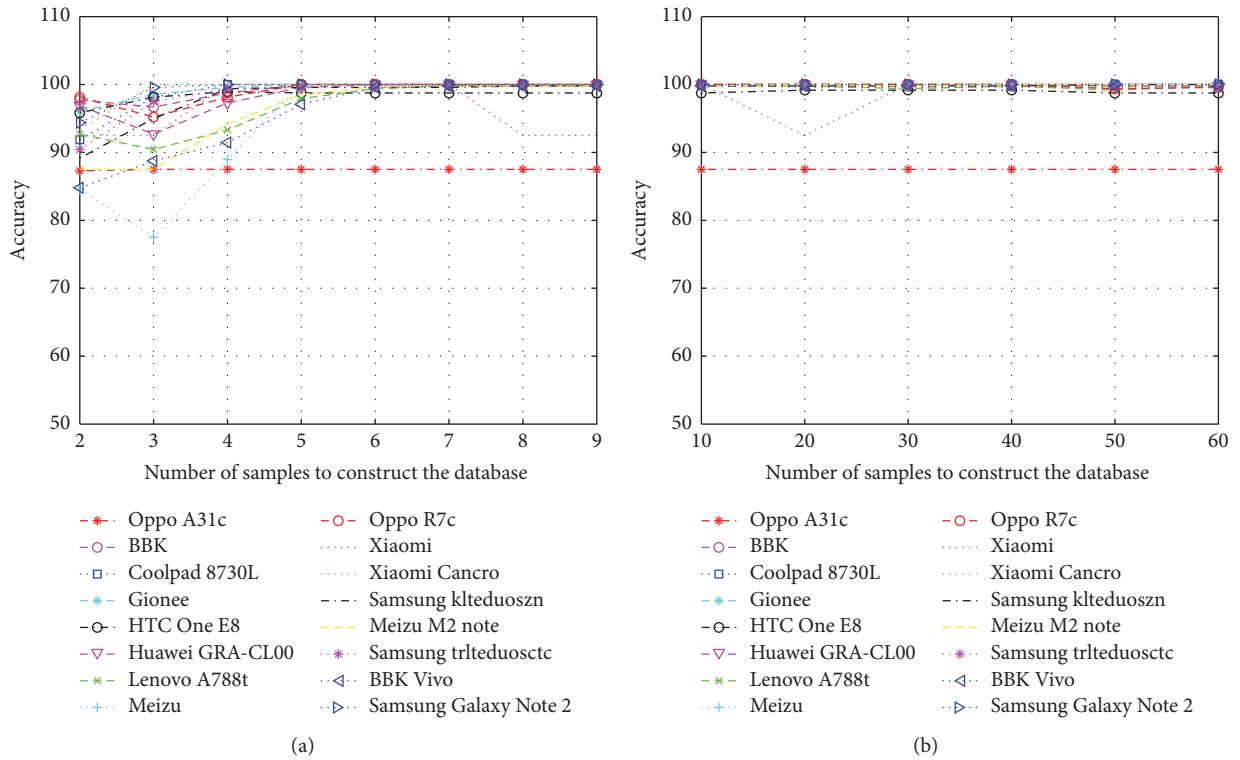



Figure 5: Localization accuracy using the fully normalized rank based on linear SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

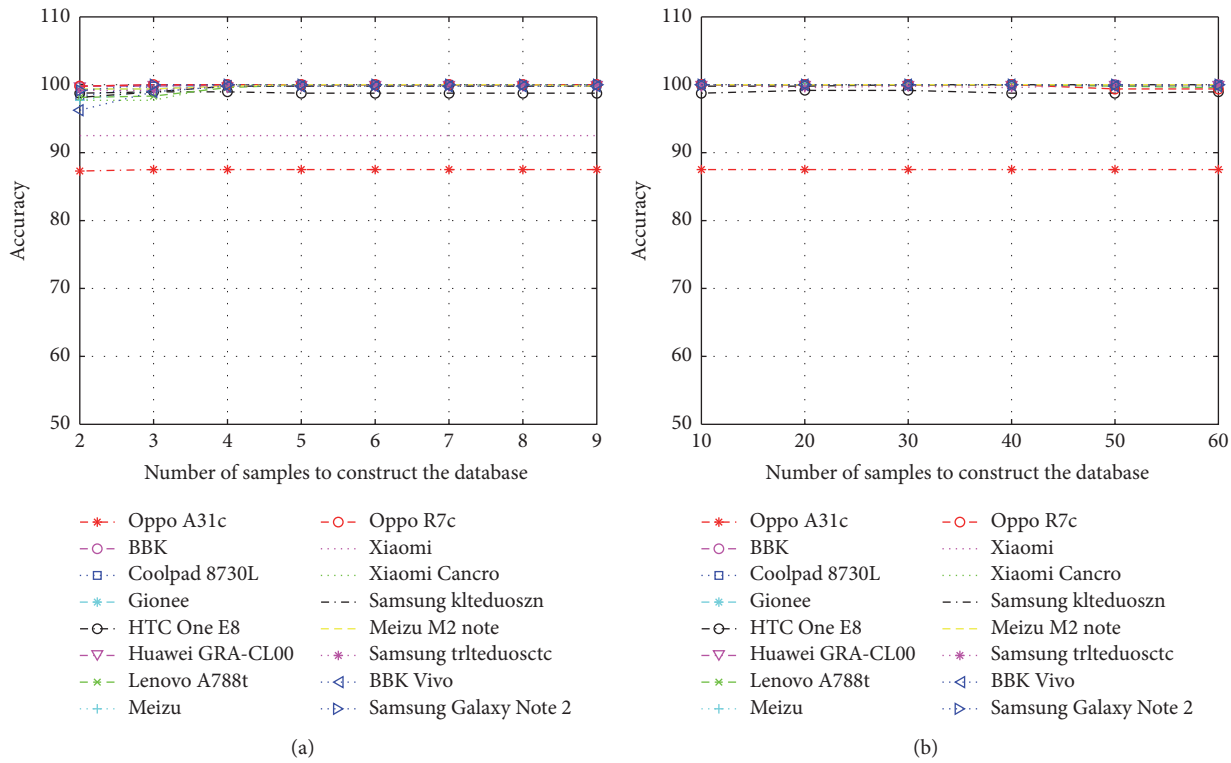



Figure 6: Localization accuracy using the vector normalized rank based on linear SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

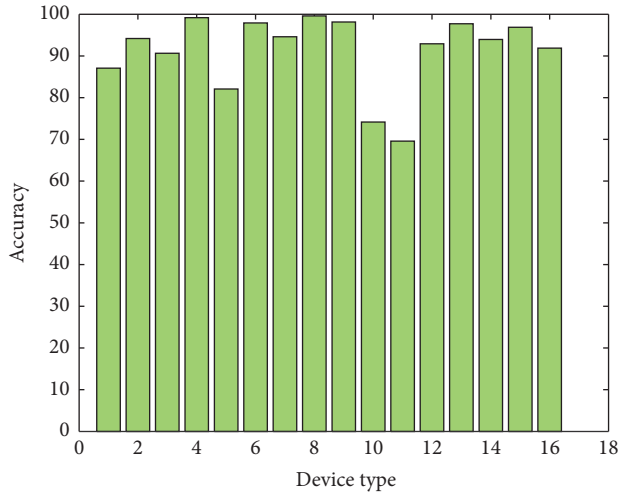
case of *Xiaomi* device that reached its maximum until the consideration of the 10th sample. This approach allows considering the varying range of the allocated tags in the new formulated vectors at one position, and consequently the difference between vectors dimensionalities. In contrast to the previous normalization, in the case of $\lambda = 1/\Psi_M$, it does not only perform as a scaling factor but also puts the transformed values of both real location fingerprint vector and the runtime vector as close as possible.

*5.4. Experiments Based on a Unique Set of Observations.* The experiments in this scenario consider about one sample designating a special Reference Point. In the test, the stored fingerprints in the radio map are attained by averaging the whole recorded dataset. Figure 7 illustrates the percentage of the correct location determination based on a small size database using the linear SVM classification. Figures 7(a), 7(b), and 7(c) are, respectively, obtained by accounting for the absolute RSSI value, the inverse rank value, and the rank value. Finally, Figure 7(d) shows the performance of the learning method by applying the proposed method. The aim of comparing between Figures 7(b) and 7(c) is to demonstrate the importance of considering the throughput signal of an input variable in the right direction. As notable in Figure 7(b), a wrong direction leads to decreasing the accuracy comparing with the reference Figure 7(a) based on RSSI value.

On the one hand, the allocation of tags in the right order results in providing the APs with a high received signal strength by high ranks, while the less reliable ones are assigned the low values. On the other hand, this will result in the prioritization of the principal features by according minor consideration to the APs with low signal strength when building the support vectors. This is due to the fact that the decision boundaries of the support vectors of SVM are always dominated by the large quantities. The built model, in this case, is much fitting the needs of this study than the inverse rank transformation. Moreover, a significant improvement is apparent by implementing the proposed solution with no less than 98.75% accurate estimation in 93.75% of the tested cases and 100% accuracy in 56.25% of cases.

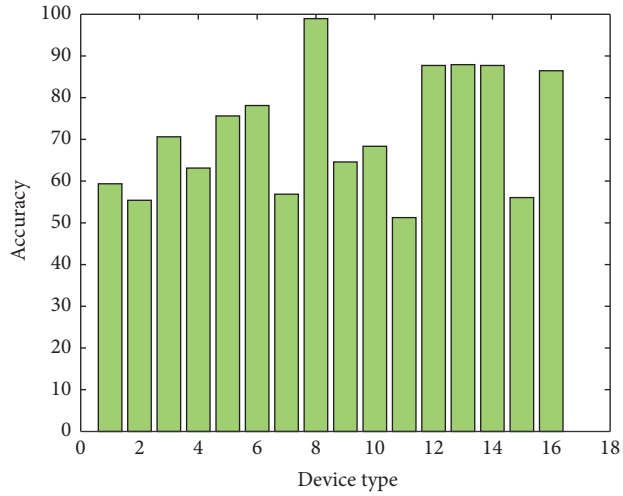
Now turning attention to *Xiaomi* device, it turns out that fusing data reduces the effect of the large variation in the registered signal strength and enhances the prediction performance. We attempted to investigate the volume of needed data to be fused to consider one fingerprint for a given Reference Point (RP). From Figure 8, it appears that both ways of normalization herein work better than just applying the rank transformation with more stable prediction. Quite similar performances are seen in Figures 8(b) and 8(c) increasing by the rise of combined information, comparing with Figure 8(a).

*5.5. Algorithms Comparison.* This section is dedicated to the comparison and the evaluation of the effectiveness of SVM among multiple ML techniques within a single experimental environment. To validate our proposed method, we



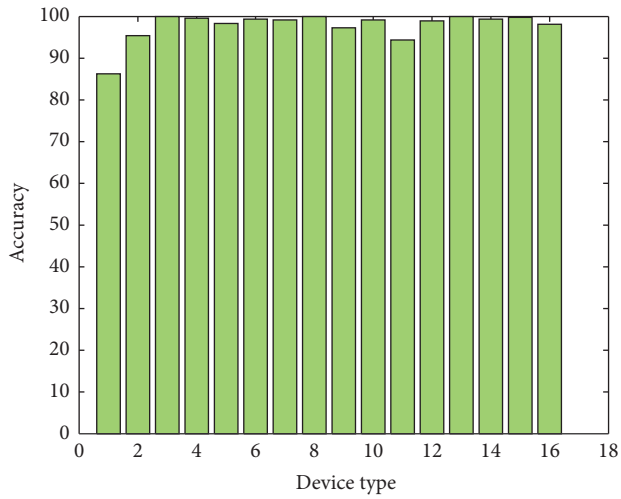

(1) Oppo A31c
(2) BBK
(3) Coolpad 8730L
(4) Gionee
(5) HTC One E8
(6) Huawei GRA-CL00
(7) Lenovo A788t
(8) Meizu
(9) Oppo R7c
(10) Xiaomi
(11) Xiaomi Cancro
(12) Samsung klteduoszn
(13) Meizu M2 note
(14) Samsung trlteduosctc
(15) BBK Vivo
(16) Samsung Galaxy Note 2

(a)



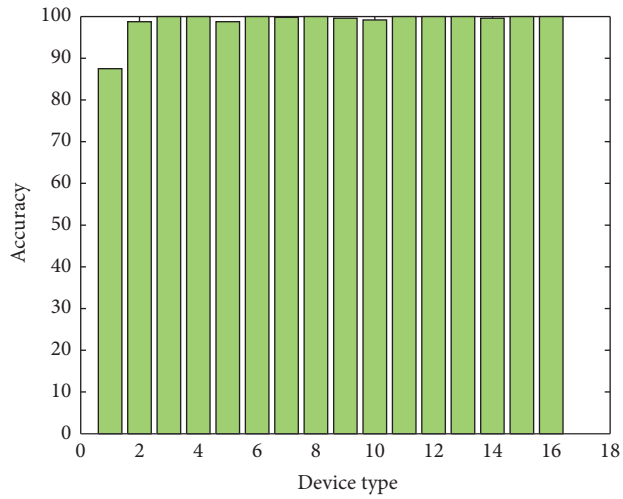

(1) Oppo A31c
(2) BBK
(3) Coolpad 8730L
(4) Gionee
(5) HTC One E8
(6) Huawei GRA-CL00
(7) Lenovo A788t
(8) Meizu
(9) Oppo R7c
(10) Xiaomi
(11) Xiaomi Cancro
(12) Samsung klteduoszn
(13) Meizu M2 note
(14) Samsung trlteduosctc
(15) BBK Vivo
(16) Samsung Galaxy Note 2

(b) #




(1) Oppo A31c
(2) BBK
(3) Coolpad 8730L
(4) Gionee
(5) HTC One E8
(6) Huawei GRA-CL00
(7) Lenovo A788t
(8) Meizu
(9) Oppo R7c
(10) Xiaomi
(11) Xiaomi Cancro
(12) Samsung klteduoszn
(13) Meizu M2 note
(14) Samsung trlteduosctc
(15) BBK Vivo
(16) Samsung Galaxy Note 2

(c)




(1) Oppo A31c
(2) BBK
(3) Coolpad 8730L
(4) Gionee
(5) HTC One E8
(6) Huawei GRA-CL00
(7) Lenovo A788t
(8) Meizu
(9) Oppo R7c
(10) Xiaomi
(11) Xiaomi Cancro
(12) Samsung klteduoszn
(13) Meizu M2 note
(14) Samsung trlteduosctc
(15) BBK Vivo
(16) Samsung Galaxy Note 2

(d)

Figure 7: Localization accuracy comparison in the case of unique exemplification per location based on linear SVM. (a) shows the attained accuracy based on RSSI value. (b) shows the accuracy in the case of applying the inverse rank value. (c) shows the accuracy in the case of applying the rank transformation value. (d) Accuracy based on the normalized rank transformation. (#) The inverse rank is the converted absolute RSSI vector by considering the opposite direction of (3).

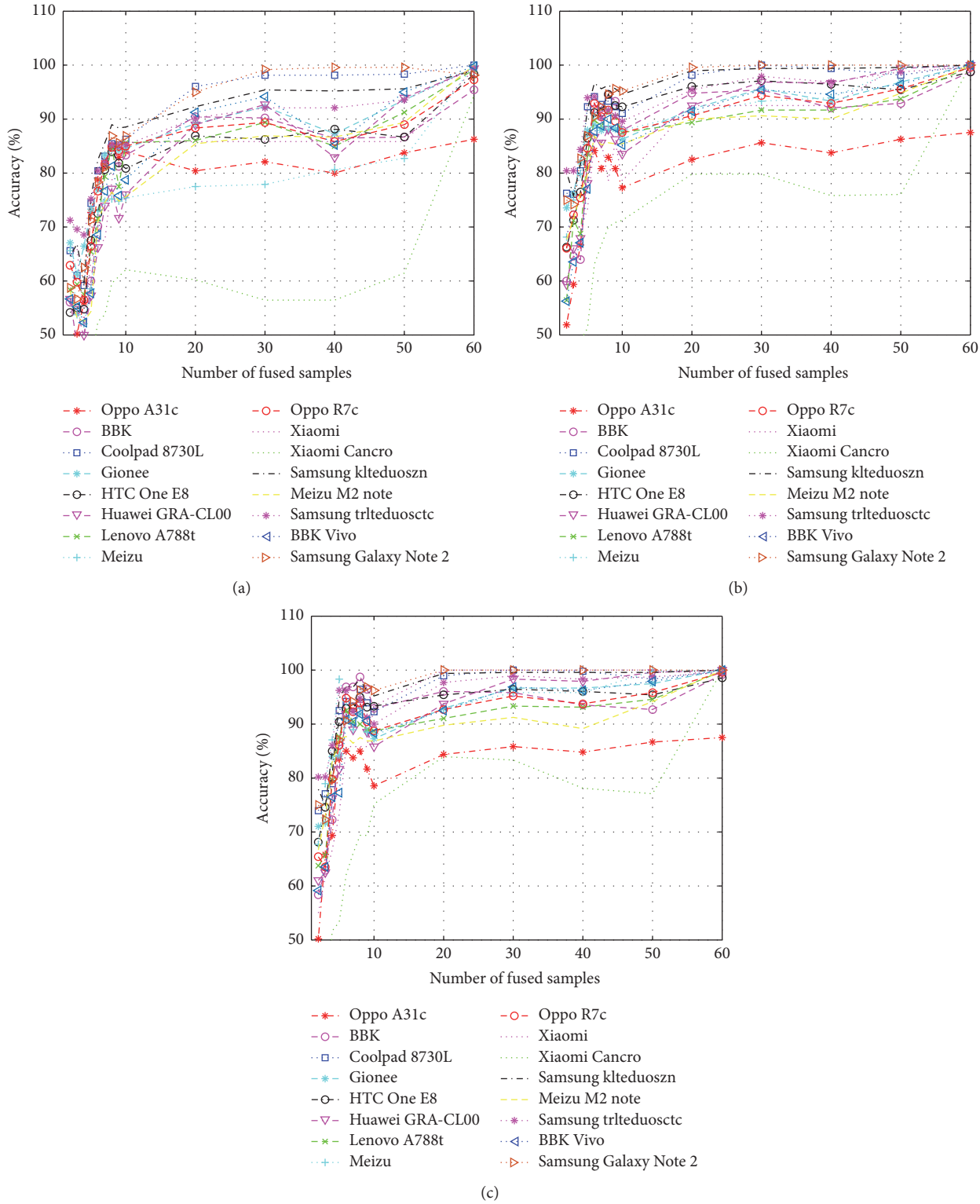



Figure 8: Localization accuracy in the case of unique exemplification per location based on linear SVM considering different data fusion size. (a) shows the attained accuracy based on the rank value. (b) shows the accuracy in the case of applying the proposed rank transformation. (c) shows the accuracy in the case of applying the fully normalized rank transformation.

Table 1: Percentage of accurately estimated locations based on linear SVM adopting multiple combinations of constraints violation along with comparison between RBF kernel and regularized Logistic Regression; the radio map is defined upon RSSI values.

| Devices | Classification type | | | | | | |
|---|---|---|---|---|---|---|---|
| | L2-R* L2-N (primal) | L2-R L2-N (dual) | L2-R L1-N (dual) | L2 LR (primal) | L1-R L2-N** | L1 LR*** | RBF kernel |
| Oppo A31c | **87.08%** | **85.42%** | **85.42%** | **86.04%** | **52.29%** | **58.75%** | **37.29%** |
| BBK | 94.17% | 95.20% | 95.20% | 95.42% | **61.45%** | **70.20%** | **35.00%** |
| Coolpad 8730L | 90.62% | **89.78%** | **89.78%** | 90.42% | **70.42%** | **76.04%** | **41.45%** |
| Gionee | 99.17% | 99.37% | 99.37% | 99.37% | **58.12%** | **73.12%** | **37.29%** |
| HTC One E8 | **82.08%** | **81.45%** | **81.45%** | **81.87%** | **54.58%** | **54.17%** | **36.04%** |
| Huawei GRA-CL00 | 97.92% | 98.12% | 98.12% | 98.33% | **63.75%** | **64.58%** | **28.75%** |
| Lenovo A788t | 94.58% | 94.78% | 94.78% | 95.00% | **51.04%** | **66.25%** | **26.67%** |
| Meizu | 99.58% | 99.58% | 99.58% | 99.58% | **88.54%** | **87.29%** | **52.70%** |
| Oppo R7c | 98.12% | 98.12% | 98.12% | 97.92% | **55.62%** | **73.33%** | **36.45%** |
| Xiaomi | **74.17%** | **74.17%** | **74.17%** | **74.17%** | **40.83%** | **63.95%** | **37.50%** |
| Xiaomi Cancro | **69.58%** | **68.54%** | **68.54%** | **69.58%** | **37.50%** | **42.29%** | **32.91%** |
| Samsung klteduoszn | 92.92% | 92.50% | 92.50% | 93.12% | **68.54%** | **63.33%** | **47.50%** |
| Meizu M2 note | 97.71% | 97.08% | 97.08% | 98.33% | **83.75%** | **79.37%** | **58.12%** |
| Samsung trlteduosctc | 93.95% | 93.75% | 93.75% | 93.96% | **72.50%** | **66.45%** | **41.67%** |
| BBK Vivo | 96.87% | 97.08% | 97.08% | 97.29% | **57.91%** | **67.92%** | **30.20%** |
| *Standard device accuracy* | 91.87% | 91.87% | 91.87% | 92.29% | **79.79%** | **78.95%** | **54.79%** |
| *Recall (ratio)* | 0.92 | 0.92 | 0.92 | 0.97 | 0.79 | 0.79 | 0.55 |
| *Precision (ratio)* | 0.95 | 0.95 | 0.95 | 0.97 | 0.86 | 0.83 | 0.77 |

(∗) R stands for regularization; (∗∗) N stands for norm; (∗∗∗) LR: stands for logistic regression. We put in bold the percentages lower than 90% for easiness of observation.

Table 2: Algorithms comparison based on the percentage of the well-recognized positions; the radio map defined upon RSSI values.

| Device type | Classification type | | | | |
|---|---|---|---|---|---|
| | SVM L2-R L2-N | Artificial Neural Network | Naïve Bayes | Random Forest | KNN ($K = 1$) |
| Oppo A31c | **87.08%** | 97.50% | **77.50%** | **59.58%** | **46.04%** |
| BBK | 94.17% | **56.25%** | **33.75%** | **59.38%** | **46.67%** |
| Coolpad 8730L | 90.62% | 97.71% | **64.38%** | **80.00%** | **81.46%** |
| Gionee | 99.17% | **74.17%** | **37.71%** | **69.38%** | **58.96%** |
| HTC One E8 | **82.08%** | 91.88% | **56.88%** | **62.71%** | **81.04%** |
| Huawei GRA-CL00 | 97.92% | 93.33% | **46.67%** | **66.04%** | **73.12%** |
| Lenovo A788t | 94.58% | **78.75%** | **35.83%** | **61.88%** | **53.33%** |
| Meizu | 99.58% | 99.79% | 95.00% | **84.00%** | 99.58% |
| Oppo R7c | 98.12% | **62.92%** | **36.04%** | **63.75%** | **50.83%** |
| Xiaomi | **74.17%** | **81.04%** | **55.00%** | **56.46%** | **78.12%** |
| Xiaomi Cancro | **69.58%** | **63.75%** | **38.75%** | **44.17%** | **52.92%** |
| Samsung klteduoszn | 92.92% | 98.75% | **67.29%** | **74.00%** | **87.08%** |
| Meizu M2 note | 97.71% | 97.71% | **86.25%** | **74.38%** | 94.37% |
| Samsung trlteduosctc | 93.95% | 97.29% | **65.83%** | **68.96%** | **89.17%** |
| BBK Vivo | 96.87% | **75.00%** | **39.17%** | **75.00%** | **52.29%** |
| *Standard device accuracy* | 91.87% | 97.50% | **77.50%** | **84.79%** | **87.08%** |
| *Recall (ratio)* | 0.92 | 0.97 | 0.77 | 0.85 | 0.87 |
| *Precision (ratio)* | 0.95 | 0.97 | 0.85 | 0.93 | 0.93 |

We put in bold the percentages lower than 90% for easiness of observation.

compared the resulting estimation of the considered ML techniques with their ground-truth locations. Taking into account the importance of using a common database and the same way of data definition as considered in the offline phase, the conducted analysis is partitioned to two main tests. The first part is devoted to the tests based on RSSI value, while the second one is devoted to the proposed normalized rank transformation.

*5.5.1. Algorithms Comparison upon RSSI Value.* Both Tables 1 and 2 have been obtained by considering the absolute RSSI value in the calibration and the positioning phase. Table 1 shows the influence on accuracy by changing the considered regularization norm, the selected kernel for SVM classifier, and using the regularized logistic regression. Herein, for SVM classifier, the linear kernel yielded better results than the RBF kernel which achieved only 54.79% correct prediction, by

Table 3: Percentage of accurately estimated locations based on linear SVM adopting multiple combinations of constraints violation along with comparison between RBF kernel and regularized Logistic Regression; the radio map is defined upon the proposed normalized rank transformation.

| Device type | Classification type | | | | | | |
|---|---|---|---|---|---|---|---|
| | L2-R* L2-N (primal) | L2-R L2-N (dual) | L2-R L1-N (dual) | L2 LR (primal) | L1-R L2-N** | L1 LR*** | RBF kernel |
| Oppo A31c | **87.50%** | **87.50%** | **87.50%** | **87.50%** | **58.95%** | **85.83%** | **85.20%** |
| BBK | 98.75% | 98.75% | 98.75% | 99.17% | **63.12%** | 95.00% | **81.46%** |
| Coolpad 8730L | 100% | 100% | 100% | 100% | 96.87% | 99.58% | 99.79% |
| Gionee | 100% | 100% | 100% | 100% | **71.87%** | 96.04% | 94.79% |
| HTC One E8 | 98.75% | 98.75% | 98.75% | 98.75% | **87.50%** | 91.04% | 95.00% |
| Huawei GRA-CL00 | 100% | 100% | 100% | 100% | **74.37%** | 98.12% | 97.08% |
| Lenovo A788t | 99.79% | 99.79% | 99.79% | 100% | **74.37%** | 96.87% | 98.54% |
| Meizu | 100% | 100% | 100% | 100% | 94.58% | 100% | 100% |
| Oppo R7c | 99.58% | 99.58% | 99.58% | 100% | **70.62%** | 95.62% | **87.92%** |
| Xiaomi | 99.17% | 99.17% | 99.17% | 99.17% | **76.46%** | **80.20%** | **85.41%** |
| Xiaomi Cancro | 100% | 100% | 100% | 100% | **74.37%** | 91.25% | **83.33%** |
| Samsung klteduoszn | 100% | 100% | 100% | 100% | **88.75%** | 97.29% | 98.75% |
| Meizu M2 note | 100% | 100% | 100% | 100% | 91.25% | 94.37% | 100% |
| Samsung trlteduosctc | 99.58% | 99.58% | 99.58% | 99.58% | 92.08% | 95.83% | 98.95% |
| BBK Vivo | 100% | 100% | 100% | 100% | **70.83%** | 93.95% | 98.33% |
| *Standard device accuracy* | 100% | 100% | 100% | 100% | 94.79% | 99.37% | 97.50% |
| *Recall (ratio)* | 1 | 1 | 1 | 1 | 0.94 | 0.99 | 0.97 |
| *Precision (ratio)* | 1 | 1 | 1 | 1 | 0.91 | 0.99 | 0.98 |

(∗) R stands for regularization; (∗∗) N stands for norm; (∗∗∗) LR stands for logistic regression. We put in bold the percentages lower than 90% for easiness of observation.

fitting the parameters from our testbed, which was expected as stated in Cover's theorem.

It is noticeable that recognition rates and the generalization ability of the test data by L2-SVM tend to be better than those by L1-SVM. Moreover, the differences in the performance of L2-SVM applying either the dual or primal optimization problem are imperceptible. This means that L2 regularization is estimating the violating variables more conservatively than L1 regularization avoiding overfitting issue of the training samples. L1 regularization SVM tends to pick only a few variables in the case of the existence of several highly correlated variables. In addition, the number of selected variables is upper bounded by the size of the training data. However, L2 norm keeps the correlated variable by shrinking their corresponding coefficients.

It is noteworthy that the L2 regularized logistic regression behaves quite similarly to L2 regularized support vector since both of them tend to maximize the margin as reducing the loss although their considered loss is different. It is perceptible in a few cases that L2 regularized logistic regression produces slightly more accurate prediction compared to L2 regularized SVM. This is due to the fact that logistic regression is modeling probabilities and therefore some errors are calculated even for correctly classified training examples while L2-SVM does not. SVM classifier is purely discriminative and designed to give a binary classification, while the regularized logistic regression estimates the a posteriori probability of class membership. In this work, we desire a binary classification and consequently identify the decision boundary directly rather than estimate the probability of class membership. On this basis we will look at the results of L2 regularized support vector instead of the L2 regularized logistic regression on the upcoming comparison.

Analysis using RSSI value based on linear SVM gives an acceptable result in most cases. Nonetheless, the model generalization to manifold devices could facilely fail to achieve a very high precision; for instance, a low correct rate estimation has been registered with *Oppo A31c, HTC, Xiaomi,* and *Xiaomi Cancro* with, respectively, a rate of 87.08%, 82.08%, 74.17%, and 69.58%. Table 2 illustrates the performance of the optimized algorithms, in which it can be seen that the built model of Random Forest, KNN, and Naïve Bayes failed dramatically, notably with heterogeneous devices. Good performances using Neural Network and SVM are noteworthy; however, SVM outperforms the whole implemented ML algorithms.

*5.5.2. Algorithms Comparison upon the Proposed Normalized Rank Transformation.* Tables 3 and 4 show the results of the proposed normalized rank transformation in both calibration and positioning phase. We investigated its usefulness considering regularization across LR and SVM classifier for linear and RBF kernels as depicted in Table 3, as well as its suitability of operation based on several ML algorithms as presented in Table 4. In general, we noticed the same previous observation based on RSSI analysis regarding the regularization and kernel type. However, a significant improvement is perceived moving from the RSSI value consideration to the normalized

Table 4: Algorithms comparison based on the percentage of the well-recognized positions; the radio map is defined upon the proposed normalized rank transformation.

| Device type | Classification type | | | | |
|---|---|---|---|---|---|
| | SVM L2-R | Artificial Neural Network | Naïve Bayes | Random Forest | KNN ($K = 1$) |
| Oppo A31c | **87.50%** | **86.67%** | **59.38%** | **57.29%** | **54.79%** |
| BBK | 98.75% | **89.58%** | **64.38%** | **47.08%** | **64.58%** |
| Coolpad 8730L | 100% | 100% | 95.83% | **82.08%** | **86.67%** |
| Gionee | 100% | 93.54% | **62.08%** | **52.50%** | **67.29%** |
| HTC One E8 | 98.75% | 98.54% | 96.25% | **80.42%** | **83.12%** |
| Huawei GRA-CL00 | 100% | 98.54% | **71.25%** | **70%** | **81.04%** |
| Lenovo A788t | 99.79% | 96.67% | **64.38%** | **60.83%** | **77.29%** |
| Meizu | 100% | 100% | 100% | 93.54% | 99.37% |
| Oppo R7c | 99.58% | 93.54% | **66.88%** | **61.67%** | **65.41%** |
| Xiaomi | 99.17% | **83.96%** | **86.04%** | **76.88%** | **80.83%** |
| Xiaomi Cancro | 100% | 93.54% | **78.75%** | **63.54%** | **48.12%** |
| Samsung klteduoszn | 100% | 99.79% | 99.58% | **86.46%** | **87.50%** |
| Meizu M2 note | 100% | 98.75% | 99.58% | **86.46%** | 92.50% |
| Samsung trlteduosctc | 99.58% | 99.17% | 99.79% | **85.42%** | **83.75%** |
| BBK Vivo | 100% | 96.46% | 69.38% | 63.96% | 70.83% |
| *Standard device accuracy* | 100% | 100% | 100% | **89.38%** | 90.83% |
| *Recall (ratio)* | 1 | 1 | 1 | 0.89 | 0.91 |
| *Precision (ratio)* | 1 | 1 | 1 | 0.94 | 0.94 |

We put in bold the percentages lower than 90% for easiness of observation.

Table 5: Confusion matrix of *OPPO A31c* device.

| | Shop 1 | Shop 2 | Shop 3 | Shop 4 | Shop 5 | Shop 6 | Shop 7 | Shop 8 |
|---|---|---|---|---|---|---|---|---|
| Shop 1 | 60 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Shop 2 | 0 | 60 | 0 | 0 | 0 | 0 | 0 | 0 |
| Shop 3 | 0 | 0 | 60 | 0 | 0 | 0 | 0 | 0 |
| Shop 4 | 0 | 0 | 0 | 60 | 0 | 0 | 0 | 0 |
| Shop 5 | 0 | 0 | 0 | 0 | 60 | 0 | 0 | 0 |
| Shop 6 | 0 | 0 | 0 | 0 | 0 | 60 | 0 | 0 |
| Shop 7 | 0 | 0 | 0 | 0 | 0 | 0 | 60 | 0 |
| Shop 8 | 60 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

rank transformation. The converted absolute RSSI values seem to be more stable and reliable rather than their initial values. Analyzing the performance of the aforementioned ML algorithms, the accuracy based on the same device is ameliorated while the generalization of the model to diversiform devices has been outperformed based on NR-SVM. This solution shows its significance improving the optimized accuracy of SVM around the decision boundaries, where 100% perfect prediction is achieved on the same device based linear L2-SVM. Moreover, no less than 98.75% well-estimated locations are attained for the remaining devices, except the special case of *Oppo A31c*.

It is interesting to further examine the positioning error percentile for the special case of *Oppo A31c* where only 87.5% correct estimation has been recorded. We have drawn its corresponding confusion matrix given in Table 5 to summarize the performance of the classification algorithm for this device. The vertical shop specification corresponds to the actual location while the horizontal ones are the predicted shops. The diagonal part of the matrix delineates the correctly classified instance where sixty samples are defined in each shop for evaluation. It is observed that both locations "shop 1" and "shop 8" are highly confused where all vectors of shop 8 were classified as location "shop 1." In the case of having a high similarity between RSSI vectors in different locations, taking into account neither the initial RSSI values nor their converted NR values can significantly distinguish between them and provide accurate estimation.

*5.6. RSSI Transformations Comparison Based on SVM.* Several studies have proposed methods to improve the robustness of positioning systems against device diversity. Calibration-free approaches have introduced several ways of RSSI reformulation. This performance comparison is aiming to show the effectiveness of the proposed NR-SVM compared with the Hyperbolic Location Fingerprinting (HLF), the Signal Strength Differences (SSD), and the DIFF method based on SVM. The difference of signal strength between pairs of APs, namely, DIFF, was proposed in [25] to reduce the effect of diversity in devices. The main difference between

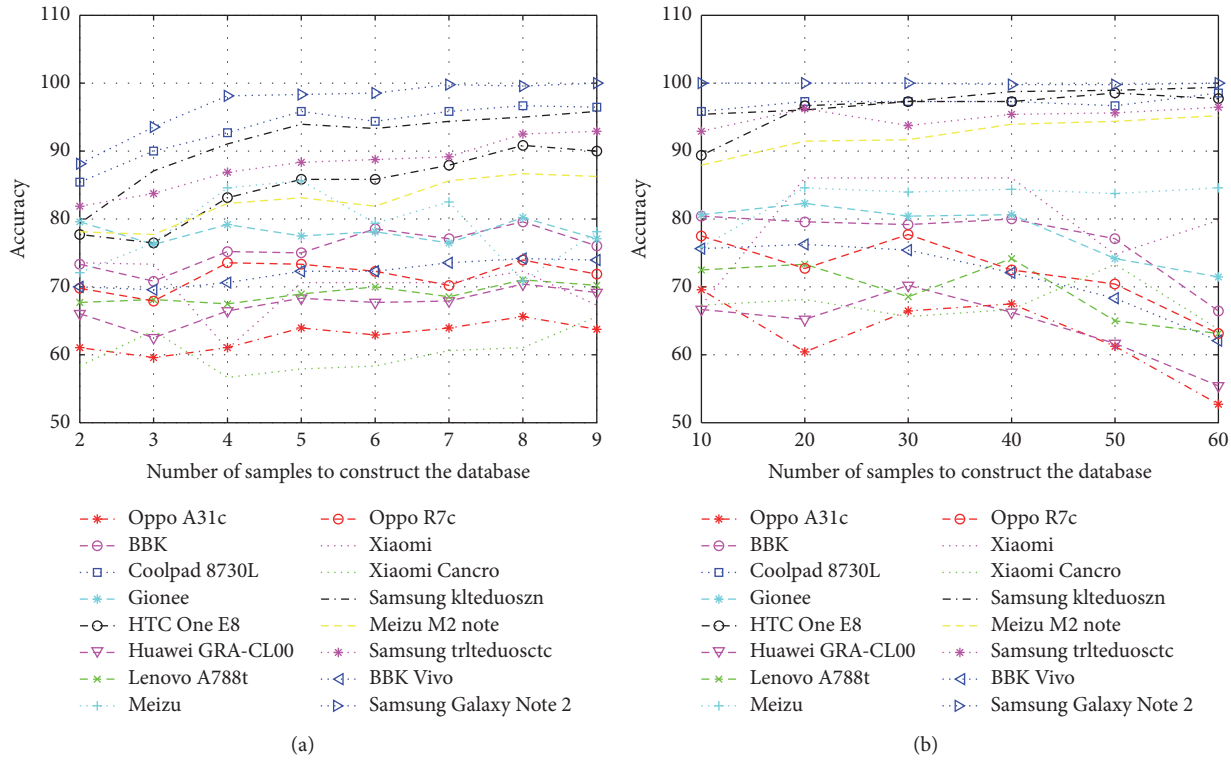



Figure 9: Achieved accuracy based on SSD-SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

the fundamental fingerprinting method and DIFF is that the latest uses the difference between pairs of APs instead of the use of the absolute RSSI value. SSD [26] takes also the advantages of Signal Strength Differences; however, it selects only independent sets of DIFF. Moreover, the HLF [24] method uses the signal strength ratios between pairs of RSSIs as fingerprints.

We implemented the aforestated RSSI value transformations and estimated the location using SVM classifier. The results show that both DIFF (Figure 10) and HLF (Figure 11) conversions perform better than the initial RSSI values (Figure 3) either when using a few samples or by considering extrasite survey measurements to identify each location. It appears in our case of study that the SSD transformation achieves lower estimation in terms of accuracy comparing the estimated positions with their ground-truth locations. Focusing on the obtained curve using the same device versus a few samples Figure 9(a), the performance is clearly below that of the absolute RSSI values. The side effect of this approach is that the performance is not always better than the initial RSSI value or even worse with homogeneous devices notwithstanding the enlargement of training samples. To further investigate this observation, we carried out the test of HLF method based on independent sets only as shown in Figure 12. It turns out that this way of transformation based on independent sets yields a significant loss of some discriminative information which decreases the accuracy instead of improving it.

NR-SVM (Figure 6) performs the best and the experiments show a consistent result as well as a high precision comparing with three of the popular calibration-free techniques DIFF, SSD, and HLF based on SVM. The reached steady performance during the whole tests proves the feasibility of considering few samples even in the case of manifold devices. Consequently, it reduces the workload of the offline data training phase. Furthermore, it avoids the time-consuming site survey in which a large number of observations are collected to boost the reliability of the system.

## 6. Conclusion

In this paper, we have presented the normalized rank transformation method for a room level determination. The performance evaluation shows that is more efficient to redefine and reformulate the signal strength observed from the anchors adequately rather than to use the absolute RSSI values from an anchor node. The importance of ranking direction and the way of normalization have been compared. The normalization factor based on the highest assigned rank results in meaningful consideration of vector dimension yielding higher accuracy, and it is characterized by its fast convergence to the maximum possible accuracy.

We explore the efficiency of NR-SVM by formulating three of the most popular calibration-free transformations proposed in the literature. DIFF, SSD, and HLF are implemented based on SVM classifier. It has been concluded that

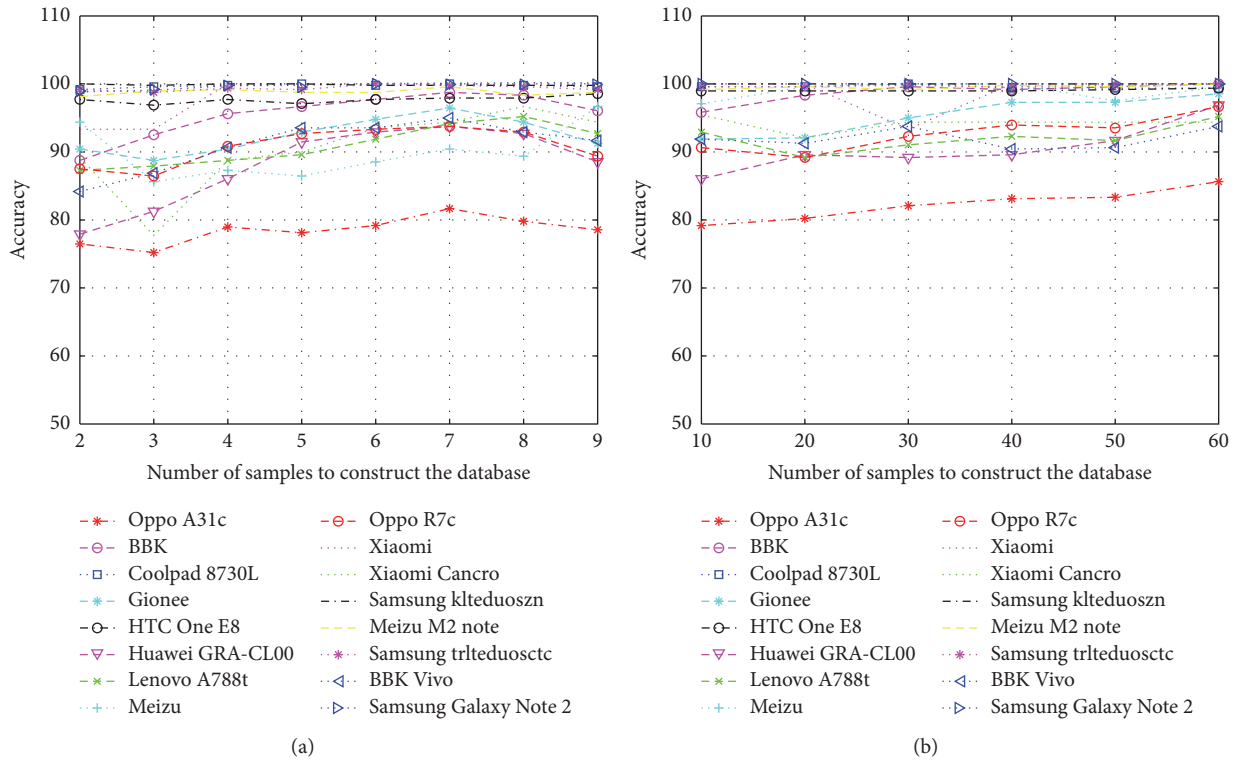



FIGURE 10: Achieved accuracy based on DIFF-SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

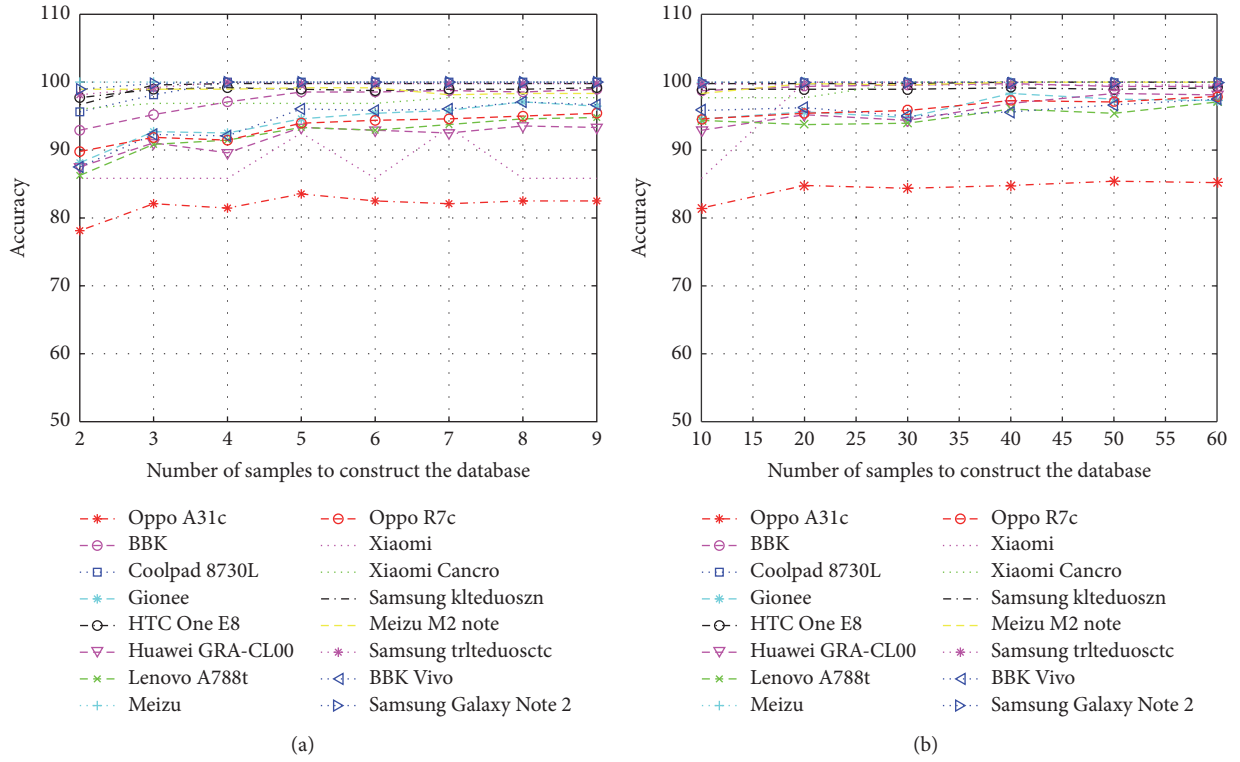



FIGURE 11: Achieved accuracy based on HLF-SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

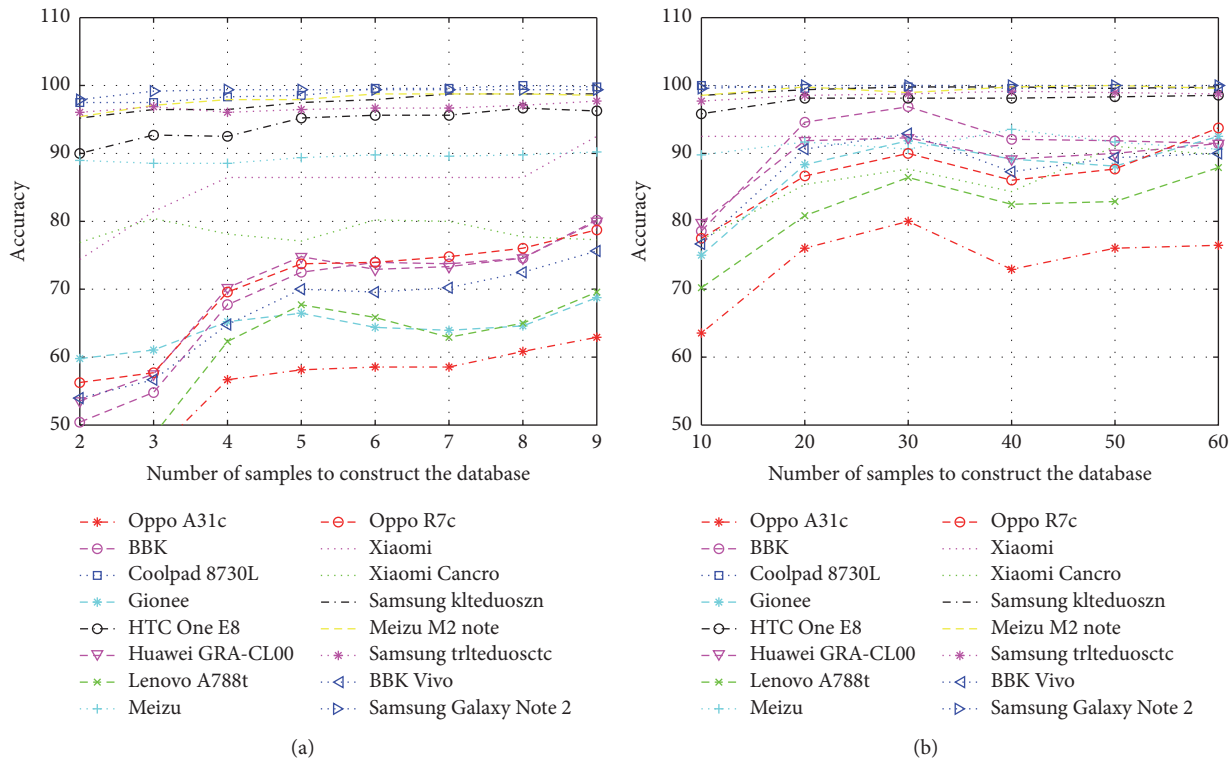



FIGURE 12: Achieved accuracy based on independent HLF-SVM. (a) shows the accuracy based on a few samples. (b) shows the accuracy based on large samples.

SSD transformation may result in a significant loss of some discriminative information and decreases the precision of the algorithm. However, both DIFF and HLF improved the performance compared to the fundamental fingerprinting method based on RSSI values. Moreover, NR-SVM has shown its distinguishability and reliability among the studied transformations.

Another contribution presented in this paper is the expansion of the application of the prebuild radio map upon a recorded information via a single device to manifold devices, taking into account the quality and stability over time of the resulting outcomes. Finally, we investigate and compare our method based on different machine learning algorithms and we showed that the linear L2 regularization norm of the Support Vector Machine outperformed Naïve Bayes, Random Forest, Artificial Neural Network, and KNN algorithms with moderate dataset in terms of accuracy, especially on handling device heterogeneity.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This work is supported by the Shanghai Science and Technology Committee under Grant 15511105100 and the National Science and Technology Major Project under Grant GFZX0301010708.

## References

[1] J. Hightower, R. Want, and G. Borriello, "Spoton: an indoor 3D location sensing technology based on Rf signal strength," UW-CSE 00-02-02, Department of Computer Science and Engineering, University of Washington, Seattle, Wash, USA, 2000.

[2] P. Bahl and V. N. Padmanabhan, "Radar: an in-building RF-based user location and tracking system," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '00)*, pp. 775–784, Tel Aviv, Israel, March 2000.

[3] R. Chen, L. Pei, J. Liu, and H. Leppäkoski, "WLAN and bluetooth positioning in smart phones," in *Ubiquitous Positioning and Mobile Location-Based Services in Smart Phones*, pp. 44–68, 2012.

[4] L. Pei, R. Chen, J. Liu, T. Tenhunen, H. Kuusniemi, and Y. Chen, "Inquiry-based bluetooth indoor positioning via RSSI probability distributions," in *Proceedings of the 2nd International Conference on Advances in Satellite and Space Communications (SPACOMM '10)*, pp. 151–156, Athens, Greece, June 2010.

[5] L. Pei, R. Chen, J. Liu, H. Kuusniemi, T. Tenhunen, and Y. Chen, "Using inquiry-based Bluetooth RSSI probability distributions for indoor positioning," *Global Positioning Systems*, vol. 9, no. 2, pp. 122–130, 2010.

[6] G. Gomes and S. Helena, "Indoor location system using ZigBee technology," in *Proceedings of the 3rd International Conference on Sensor Technologies and Applications (SENSORCOMM '09)*, pp. 152–157, Athens, Greece, June 2009.

[7] K. Pahlavan, F. O. Akgül, M. Heidari, A. Hatami, J. M. Elwell, and R. D. Tingley, "Indoor geolocation in the absence of direct path," *IEEE Wireless Communications*, vol. 13, no. 6, pp. 50–58, 2006.

[8] J. MacHaj, P. Brida, and J. Benikovsky, "Using GSM signals for fingerprint-based indoor positioning system," in *Proceedings of the 10th International Conference (ELEKTRO '14)*, pp. 64–67, Rajecke Teplice, Slovakia, May 2014.

[9] Y. Yuan, L. Pei, C. Xu, Q. Liu, and T. Gu, "Efficient WiFi fingerprint training using semi-supervised learning," in *Proceedings of the Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS '14)*, pp. 148–155, Corpus Christ, Tex, USA, November 2014.

[10] M. Youssef and A. Agrawala, "The Horus location determination system," *Wireless Networks*, vol. 14, no. 3, pp. 357–374, 2008.

[11] K. Kaemarungsi and P. Krishnamurthy, "Properties of indoor received signal strength for WLAN location fingerprinting," in *Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS '04)*, pp. 14–23, Boston, MA, USA, August 2004.

[12] L. N. Chen, B. H. Li, K. Zhao, C. Rizos, and Z. Q. Zheng, "An improved algorithm to generate a Wi-Fi fingerprint database for indoor positioning," *Sensors*, vol. 13, no. 8, pp. 11085–11096, 2013.

[13] B. Zhao, L. Pei, C. Xu, and L. Gu, "Graph-based efficient WiFi fingerprint training using un-supervised learning," in *Proceedings of the 28th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS '15)*, pp. 2301–2310, Tampa Convention Center, Tampa, Fla, USA, September 2015.

[14] G. Shakhnarovich, P. Indyk, and T. Darrell, "Nearest-neighbor methods in learning and vision," *IEEE Transactions on Neural Networks*, vol. 19, no. 2, article 377, 2008.

[15] R. M. Faragher and R. K. Harle, "SmartSLAM—an efficient smartphone indoor positioning system exploiting machine learning and opportunistic sensing," *ION GNSS*, vol. 13, pp. 1006–1019, 2013.

[16] Mitchell and M. Tom, "Machine Learning," in *McGraw-Hill Science/Engineering/Math*, 2007, (Bayes Theorem) 156–163, (Naïve Bayes classifier) 177-178, (KNN) 230–234.

[17] L. B. Del Mundo, R. L. D. Ansay, C. A. M. Festin, and R. M. Ocampo, "A comparison of Wireless Fidelity (Wi-Fi) fingerprinting techniques," in *Proceedings of the 2011 International Conference on ICT Convergence (ICTC '11)*, pp. 20–25, Seoul, South Korea, September 2011.

[18] A. H. Salamah, M. Tamazin, M. A. Sharkas, and M. Khedr, "An enhanced WiFi indoor localization System based on machine learning," in *Proceedings of the 2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN '16)*, pp. 1–8, Alcala de Henares, Spain, October 2016.

[19] L. Calderoni, M. Ferrara, A. Franco, and D. Maio, "Indoor localization in a hospital environment using random forest classifiers," *Expert Systems with Applications*, vol. 42, no. 1, pp. 125–134, 2015.

[20] C. Zhou and A. Wieser, "Application of backpropagation neural networks to both stages of fingerprinting based WIPS," in *Proceedings of the 2016 4th International Conference on Ubiquitous Positioning, Indoor Navigation and Location Based Services (UPINLBS '16)*, pp. 207–217, Shanghai, China, November 2016.

[21] M. S. Ifthekhar, N. Saha, and Y. M. Jang, "Neural network based indoor positioning technique in optical camera communication system," in *Proceedings of the 5th International Conference on Indoor Positioning and Indoor Navigation( IPIN '14)*, pp. 431–435, Busan, South Korea, October 2014.

[22] L. Pei, M. Zhang, D. Zou, R. Chen, and Y. Chen, "A survey of crowd sensing opportunistic signals for indoor localization," *Mobile Information Systems*, vol. 2016, Article ID 4041291, 16 pages, 2016.

[23] A. W. Tsui, Y.-H. Chuang, and H.-H. Chu, "Unsupervised learning for solving RSS hardware variance problem in WiFi localization," *Mobile Networks and Applications*, vol. 14, no. 5, pp. 677–691, 2009.

[24] M. B. Kjærgaard and C. V. Munk, "Hyperbolic location fingerprinting: a calibration-free solution for handling differences in signal strength (concise contribution)," in *Proceedings of the 6th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom '08)*, Hong Kong, China, March 2008.

[25] F. Dong, Y. Chen, J. Liu, Q. Ning, and S. Piao, "A calibration-free localization solution for handling signal strength variance," in *Mobile Entity Localization and Tracking in GPS-Less Environments*, pp. 79–90, Springer, Berlin, Germany, 2009.

[26] A. K. M. Mahtab Hossain, Y. Jin, W. Soh, and H. N. Van, "SSD: a robust RF location fingerprint addressing mobile devices' heterogeneity," *IEEE Transactions on Mobile Computing*, vol. 12, no. 1, pp. 65–77, 2013.

[27] J. Correa, E. Katz, P. Collins, and M. Griss, "Room-Level WiFi Location Tracking," MRC-TR-2008-02, Carnegie Mellon Silicon Valley, CyLab Mobility Research Center, 2008.

[28] A. Buchman and C. Lung, "Received signal strength based room level accuracy indoor localisation method," in *Proceedings of the 4th IEEE International Conference on Cognitive Infocommunications (CogInfoCom '13)*, pp. 103–108, Budapest, Hungary, December 2013.

[29] S. N. Patel, K. N. Truong, and G. D. Abowd, "Powerline positioning: a practical sub-room-level indoor location system for domestic use," in *Proceedings of the 8th International Conference (UbiComp '06)*, pp. 441–458, Orange County, CA, USA, 2006.

[30] R. Yasmine and L. Pei, "Indoor fingerprinting algorithm for room level accuracy with dynamic database," in *Proceedings of the 2016 4th International Conference on Ubiquitous Positioning, Indoor Navigation and Location Based Services (UPINLBS '16)*, pp. 113–121, Shanghai, China, November 2016.

[31] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 1995.

[32] V. N. Vapnik, *Statistical Learning Theory*, Wiley, New York, NY, USA, 1998, ISBN: 978-0-471-03003-4. pp. I–XXIV, 1–736.

[33] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers*, vol. EC-14, no. 3, pp. 326–334, 1965.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 4802159, 11 pages
https://doi.org/10.1155/2017/4802159

*Research Article*

# An Online Solution of LiDAR Scan Matching Aided Inertial Navigation System for Indoor Mobile Mapping

**Xiaoji Niu, Tong Yu, Jian Tang, and Le Chang**

*GNSS Research Centre, Wuhan University, 129 Luoyu Road, Wuhan 430079, China*

Correspondence should be addressed to Jian Tang; tangjian@whu.edu.cn

Received 23 February 2017; Revised 17 May 2017; Accepted 30 May 2017; Published 3 July 2017

Academic Editor: Gonzalo Seco-Granados



Multisensors (LiDAR/IMU/CAMERA) integrated Simultaneous Location and Mapping (SLAM) technology for navigation and mobile mapping in a GNSS-denied environment, such as indoor areas, dense forests, or urban canyons, becomes a promising solution. An online (real-time) version of such system can extremely extend its applications, especially for indoor mobile mapping. However, the real-time response issue of multisensors is a big challenge for an online SLAM system, due to the different sampling frequencies and processing time of different algorithms. In this paper, an online Extended Kalman Filter (EKF) integrated algorithm of LiDAR scan matching and IMU mechanization for Unmanned Ground Vehicle (UGV) indoor navigation system is introduced. Since LiDAR scan matching is considerably more time consuming than the IMU mechanism, the real-time synchronous issue is solved via a one-step-error-state-transition method in EKF. Stationary and dynamic field tests had been performed using a UGV platform along typical corridor of office building. Compared to the traditional sequential postprocessed EKF algorithm, the proposed method can significantly mitigate the time delay of navigation outputs under the premise of guaranteeing the positioning accuracy, which can be used as an online navigation solution for indoor mobile mapping.

## 1. Introduction

The establishment of highly efficient, accurate, and low-cost indoor mapping technology is becoming more and more necessary and urgent, due to the growing interest and market of indoor Location Based Services (LBSs). Simultaneous Location and Mapping (SLAM) has become a popular and effective indoor mapping technology in recent years. The SLAM technique is a process of building a map of an unknown environment by traversing it with range sensors while determining the system location on the map simultaneously; it has been explored in robotics and computer science for decades. LiDAR-based SLAM is one of the most successful technologies, as it can provide high frequency and high precision range measurements [1]. It combines positioning and mapping by utilizing two or more consecutive frames of scan points (called scan matching) with various algorithms [2–8]. However, LiDAR-based SLAM is heavily dependent on environment features and performs poorly in a featureless area. Various data fusion solutions have been researched for a long time in order to offset the poor performance of standalone sensor [9–13]. The LiDAR/IMU integrated method is a feasible way to solve such problems [1, 5–8, 14], because inertial measurement unit- (IMU-) based inertial navigation system (INS) can offer accurate relative positions and attitudes in a short period using gyroscopes and accelerometers [5–8]. Finally, LiDAR/IMU integrated method can provide sustainable and accurate position and attitude results for indoor mobile mapping.

Presently, the most popular SLAM algorithms are divided into the following types: Kalman filter, particle filters, and graph-based. Hector SLAM estimates the 3D navigation state based on robust scan matching and inertial navigation system by Extended Kalman Filter (EKF) [15]. Gmapping is a Rao-Blackwellized Particle Filter SLAM approach which requires a high number of particles to obtain good results. Therefore, an adaptive resampling technique needs to be developed to solve the depletion problems [16]. Karto SLAM is a graph-based SLAM approach. The higher the number of landmarks, the more amount of memory required [17]. The sensor system

in this paper is only composed of LiDAR and IMU. The Extended Kalman Filter method is adopted for LiDAR/IMU integrated in order to exploit IMU's advantages in this paper.

However, most of the existing highly accurate SLAM systems are postprocessed works [18–21]. An efficient online solution is still a challenge in the field of highly accurate mapping applications. For the LiDAR/IMU integrated system, although IMU can accelerate the computation of LiDAR scan matching, the time cost of LiDAR scan matching is still much larger than the IMU sampling and processing. Thereby, in an online application, if the IMU data are received when the LiDAR scans are matching, these IMU data have to wait in the buffer until the LiDAR scan matching process is finished via the traditional sequential postprocessed method. The time delay will hinder the IMU observation in current time from being obtained and processed in time. This is a fatal influence for an online system [22].

Various methods for the time delay issue have already been addressed in previous works; many efforts concentrate on optimizing the fusion filter [23–25]. Guivant et al. presented an optimal algorithm that significantly reduces the computational requirements by propagating the stored information in the local area to the rest of the global map in only one iteration [22]. Other methods include dividing the large-scale maps into small amenable maps [24] or building a hierarchical submap method [23]; these algorithms are suitable for large-scale maps that have several landmarks. In 2016, Google's Cartographer provides a real-time solution for indoor mapping, which belongs to the graph-based SLAM algorithm. Its time consumption of LiDAR scan matching is far less than that in this paper. However, the Cartographer needs to achieve real-time loop closure to eliminate the accumulated errors and the CPU occupancy is extremely high for accelerating the computation speed [26]. In addition, optimizing a large number of variables in the SLAM issue simultaneously by using two algorithms is another alternative to do mapping in real-time. One algorithm performs odometry at a high frequency but low fidelity to estimate velocity of the LiDAR. Another algorithm runs at a frequency of an order of magnitude lower for fine matching and registration of the point cloud [27].

The details of the LiDAR/IMU fused method were introduced in our previous work [4, 5]. In this paper, we will introduce an efficient online solution based on the compensation method by state propagating for the LiDAR/IMU integrated inertial navigation system for indoor mobile mapping, which is an extension of our previous works. This state propagating method has been studied in a GNSS/IMU integrated system but not in a LiDAR/IMU integrated system. It has the advantages of low computation complexity, simple implementation, and high reliability [28].

The proposed LiDAR scan matching algorithm in this paper is an improved, probabilistically motivated Maximum Likelihood Estimation (IMLE) algorithm. It is a brute global optimum search method that can obtain the global optimum position for each scan matching. Aided with IMU, its mapping precision can achieve centimeter-level with the current postprocessed Extended Kalman Filter (EKF) method [4]. The original sequential LiDAR scan matching and IMU mechanism process workflow is divided into two independent and parallel workflows. Then, the time delay between the LiDAR scan matching and IMU mechanism is synchronized via the one-step-error-state-transition method, which is applied to a standard EKF. Compared with existing online mapping solutions, this paper offers several major contributions: (a) it presents an improved method based on our previous works [4, 5], keeping the high precision characteristics of the LiDAR/IMU fusion algorithm; (b) the parallel approach can make the LiDAR scan matching and IMU mechanization operate independently, accelerating the processing of the whole system; (c) the one-step-error-state-transition mathematic model applied in EKF only records and propagates the error state in EKF prediction epochs and does not increase the dimension of the state matrix, which keeps the data fusion synchronized and reduces the computation complexity and processing time.

The rest of this paper is organized as follows: Section 2 gives the workflow of the online LiDAR/IMU integrated system and describes the mathematic principle of the proposed method; Section 3 presents the indoor field tests and discusses the results; conclusions are drawn in Section 4.

## 2. Online LiDAR/IMU Integrated System Modeling

*2.1. Sequential IMU and LiDAR Fusion Modeling.* The overview of the traditional sequential LiDAR/IMU integrated system mathematic model is shown in Figure 1. The sampling rates of IMU and LiDAR are approximately 200 Hz and 10 Hz, respectively. The rate of IMU is higher than that of LiDAR. IMU can calculate the position ($r^n_{\mathrm{IMU}}$), velocity ($v^n_{\mathrm{IMU}}$), and attitudes ($C^n_{b\,\mathrm{IMU}}$) by using the mechanization algorithm, when no LiDAR observation information is received. The local level frame of north, east, and down (NED) named navigation frame (n-frame) is taken as the reference frame for the inertial navigation. The body frame (b-frame) is defined at the IMU's centre, with the axes pointing forward, right, and down, respectively. In fact, the IMU output contains errors and the errors will cause the navigation results to drift rapidly over a long time. Thus, an error propagation model must work alongside the system motion model to further correct and obtain better navigation results. The Phi-Angle error model is selected to describe the time-dependent behavior errors [25, 26]. The error state vector is defined in the n-frame as follows:

$$\delta x\,(t) = \left[\left(\delta r^n_{\mathrm{IMU}}\right)^T \left(\delta v^n_{\mathrm{IMU}}\right)^T \epsilon^T b_g^T b_a^T\right], \tag{1}$$

where the error state $\delta x$ consists of the errors of position ($\delta r^n_{\mathrm{IMU}}$), the errors of velocity ($\delta v^n_{\mathrm{IMU}}$), the errors of attitude ($\epsilon$), the bias of gyroscope ($b_g$), and the bias of acceleration ($b_a$), which is a 15-dimension vector.

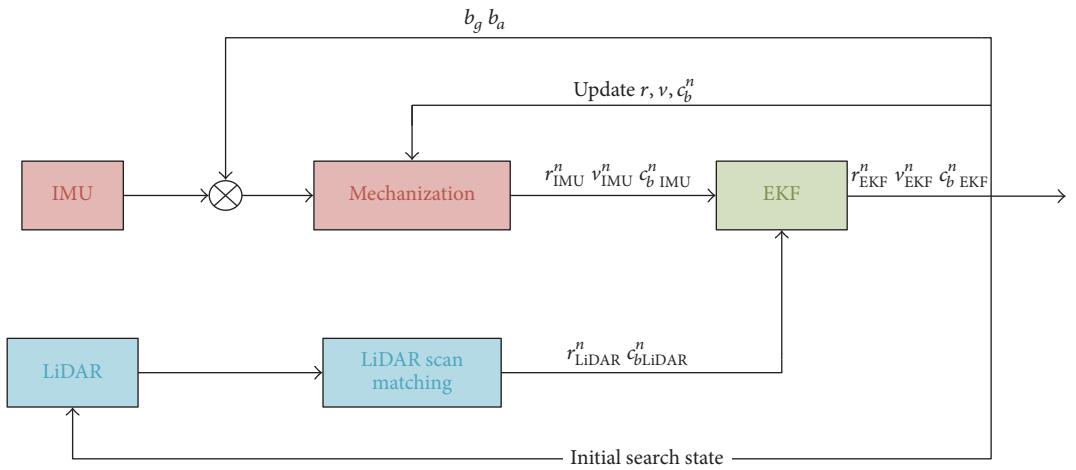



Figure 1: The mathematical model of the LiDAR/IMU integrated system.

The biases of gyroscope and accelerometer are modeled as a first-order Gauss-Markov process with correlation time $T$ and mean square value $\sigma^2$. The model is described by

$$\dot{b}_g(t) = -\frac{1}{T_{gb}} b_g(t) + w_{gb}(t),$$
$$\dot{b}_a(t) = -\frac{1}{T_{ab}} b_a(t) + w_{ab}(t). \tag{2}$$

The INS error model with the sensor error models in continuous time can be expressed by

$$\delta\dot{x}(t) = F(t)\,\delta x(t) + G(t)\,w(t). \tag{3}$$

$F$ is the dynamic matrix, $G$ is a noise-input mapping matrix, and $w$ is the forcing vector of white noise, according to the system motion model and concrete formation of $F$, $G$ that can be found in the works of Shin, 2001 and 2005 [29, 30].

The discrete form of (3) is

$$\delta x_k = \phi_{k/k-1}\delta x_{k-1} + G_{k-1}w_{k-1}, \tag{4}$$

where $\phi_{k/k-1}$ is the state transition matrix and $w_{k-1}$ is the driven white noise

$$\phi_{k/k-1} = \exp\left(F(t_k)\,\Delta t\right) \approx I + F(t_k)\,\Delta t. \tag{5}$$

$w_{k-1}$ is a sequence of zero-mean random variable and the covariance matrix associated with $w_k$ is given by

$$E\left[w_k w_j^T\right] = \begin{cases} 0, & k = j \\ Q_k, & k \neq j, \end{cases} \tag{6}$$

$$Q_k \approx \phi_k G(t_k)\, Q G(t_k)^T \phi_k^T \Delta t, \tag{7}$$

$$Q = \text{diag}\left(E\left[w_v^2\right], E\left[w_\phi^2\right], E\left[w_{gb}^2\right], E\left[w_{ab}^2\right]\right) = \text{diag}\left(\text{vrw}^2, \text{arw}^2, \frac{2\sigma_{gb}^2}{T_{gb}}, \frac{2\sigma_{ab}^2}{T_{ab}}\right). \tag{8}$$

$Q_k$ is the covariance matrix; $Q$ is the spectral density matrix; vrw and arw are velocity random walk and angular random walk, which are given by the IMU user manual; $T_{gb}$ and $T_{ab}$ are the correlation times of gyroscopes and accelerometers, respectively; $\sigma_{gb}^2$ and $\sigma_{ab}^2$ are the mean square values of gyroscopes and accelerometers, respectively, which are described in formula (2).

The LiDAR and IMU measurements are fused by the EKF algorithm only at the epoch in which LiDAR scan information is obtained. The EKF observation functions are given briefly by

$$z_k = \begin{bmatrix} r^n_{\text{IMU}} - r^n_{\text{LiDAR}} \\ \epsilon^n_{\text{IMU}} - \epsilon^n_{\text{LiDAR}} \end{bmatrix} = H_k \delta x_k + v_k, \tag{9}$$

$$H_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{10}$$

where $z_k$ is a 4-dimensional measurement vector; $r^n_{\text{IMU}}$ is the predicted position from the IMU mechanization; $r^n_{\text{LiDAR}}$ is the observed position from LiDAR; $\epsilon^n_{\text{IMU}}$ and $\epsilon^n_{\text{LiDAR}}$ are the predicted and observed heading angles, respectively, which are expressed as Euler angles. They make up the four observations. The LiDAR observations of 2-dimension position $(x, y)$ and heading angle can be obtained from the LiDAR scan matching. And owing to the flatness of building floor in indoor environment, the height of LiDAR observation is assumed constant, which is set as zero in this paper; $H_k$ is the designed matrix that describes the relation between the state vector and the measurements and is given in (10); $v_k$ is the driven response of the input white noise at time $t_{(k+1)}$.

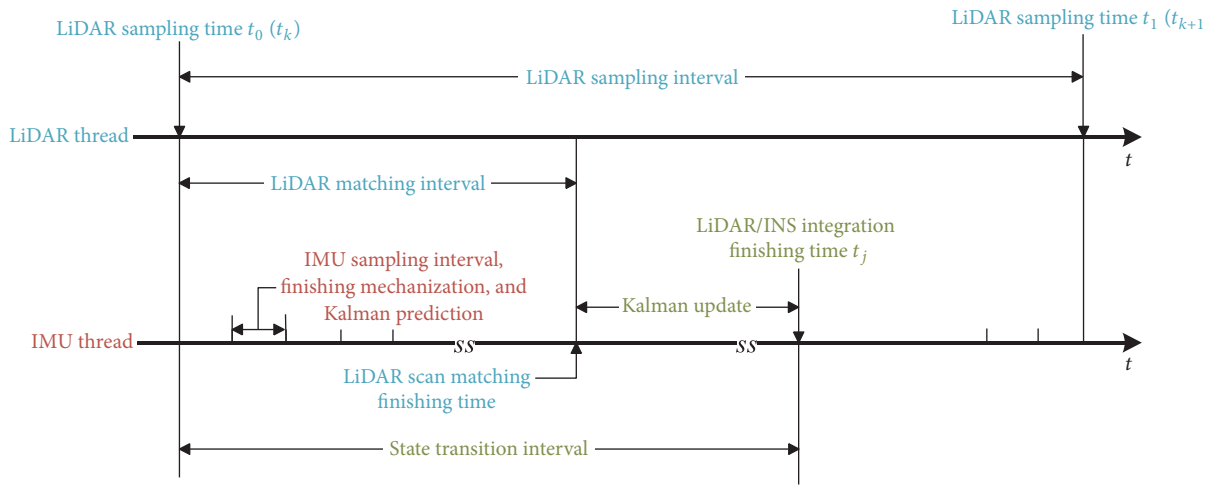



Figure 2: The parallel processing sequence diagram.

The measurement covariance matrix is written as

$$E\left[v_k v_j^T\right] = \begin{cases} 0, & k = j \\ R_k, & k \neq j \end{cases} \tag{11}$$
$$R_k = \text{diag}\left(\delta_r^2, \delta_\epsilon^2\right).$$

$R_k$ is a 4-dimension covariance matrix. $\delta_r$, $\delta_\epsilon$ are the errors of position and heading, approximate values based on the properties of the laser scanner device, and the angle and range searching intervals of the LiDAR scan matching algorithm.

The estimates of the EKF prediction functions are

$$\begin{aligned} \delta x_{k+1}^- &= \phi_k \delta x_k, \\ P_{k+1}^- &= \phi_k P_k \phi_k^T + Q_k. \end{aligned} \tag{12}$$

The Kalman gain is

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1}. \tag{13}$$

The state vector is updated as

$$\begin{aligned} \delta x_k &= \delta x_k^- + K_k \left(z_k - H_k \delta x_k^-\right), \\ P_k &= \left(I - K_k H_k\right) P_k^-, \end{aligned} \tag{14}$$

where $\delta x_k^-$ and $P_k^-$ are the prior estimate and its error covariance. The $P$ matrix, namely, the estimated standard deviations of the estimated states, consists of

$$\begin{aligned} P &= E\left\{x, x^T\right\} \\ &= \text{diag}\left(\delta_x^2, \delta_y^2, \delta_h^2, \delta_{vx}^2, \delta_{vy}^2, \delta_{vz}^2, \delta_{\varepsilon x}^2, \delta_{\varepsilon y}^2, \delta_{\varepsilon z}^2\right), \end{aligned} \tag{15}$$

where the initial value of $P$ matrix $P_0$ in the experiment is set as follows: $\delta_x$, $\delta_y$, and $\delta_z$ are the precision of initial position and are given at centimeters level, which is the precision of positioning by LiDAR scan matching; $\delta_{vx}$, $\delta_{vy}$, and $\delta_{vz}$ are small values about 0.001 m/s, because the whole system is started from stationary state [31]; $\delta_{\varepsilon x}$, $\delta_{\varepsilon y}$, and $\delta_{\varepsilon z}$ are the accuracy of initial heading, which are empirical value about 1 degree, 1 degree, and 5 degrees, respectively.

Finally, the estimated error $\delta x_k$ is fed back to the INS mechanization to correct the final output of navigation state, $r_{\text{EKF}}^n$, $v_{\text{EKF}}^n$, and $C_{b\,\text{EKF}}^n$, which will also be the initial state for the LiDAR scan matching algorithm in the next epoch. Then, the next iteration continues.

*2.2. Online Improvement.* In the previous sequential IMU and LiDAR fusion model, the next IMU mechanization must wait until the prior LiDAR scan matching has completed. For instance, with the configuration in this system, the time of LiDAR scan matching costs approximately 70 ms. This means that the following IMU data in this 70 ms cannot be processed in time. Thus, the time delay of IMU mechanization due to the processing time cost of LiDAR scan matching is a challenge for online processing.

To settle this issue, a parallel online method is utilized for LiDAR scan matching and the IMU mechanization process. Figure 2 describes the processing sequence. When scan points are received at $t_0$, LiDAR scan matching begins to work in the LiDAR thread; meanwhile, the IMU mechanization continues to run in the IMU thread. Normally, after the LiDAR scan matching finishes, the matching results have to be fused with the IMU at epoch $t_0$ in the sequential process model. However, the integration finishing time, namely, the current time, is not $t_0$. Only when the current error state estimate $\delta x$ corrects the current IMU output can we obtain the real-time information. Therefore, the calculated $\delta x_{t_0}$ and its covariance in $t_0$ have to be propagated to the current time correctly by using the one-step-error-state-transition algorithm, which will be introduced in detail in Section 2.3. Then, error state estimation $\delta x$ at the current time can be fed back to correct the final output of navigation in time.

*2.3. One-Step-Error-State-Transition.* This section will describe mathematical derivation of propagating the error state estimation of EKF from $t_0$ to the current time. The sampling frequency of IMU is 200 Hz, which means every 5 ms there will be an IMU data. Mechanization of IMU is not costly and it can be done during this 5 ms and it implies that predictions of EKF can be performed at proper times online. However, LiDAR scan matching takes much more time to get the observation result. For example, if we get IMU and LiDAR data at time $t_0$. Mechanization of IMU can get the prediction $S_{\mathrm{IMU}(t_0)}$ immediately. While LiDAR scan matching will cost about 70 ms to get the EKF observation position $S_{\mathrm{LiDAR}(t_0)}$ of time $t_0$. After scan matching, current time has shifted to $t_1 = t_0 + 70$ ms; then EKF can only update the result $S_{\mathrm{EKF}(t_0)}$ at the moment of $t_1$. At the time $t_1$, the INS prediction moved to $t_1$, and $P$ matrix and error state vector $\delta x$ have already been predicted to time $t_1$ that are $P_{t_1}^-$ and $\delta x_{t_1}^-$, which are not corrected by observation of LiDAR ($S_{\mathrm{LiDAR}(t_0)}$). Therefore, we only have the updated results $S_{\mathrm{EKF}(t_0)}$, $P_{t_0}^+$, and $\delta x_{t_0}^+$ at time $t_1$. Thereby, $P_{t_0}^+$ and $\delta x_{t_0}^+$ need to be propagated to $P_{t_1}^+$ and $\delta x_{t_1}^+$ in time for online application.

State transition means using the state in time $t_k$ to estimate the state in time $t_j$ $(j > k)$. The Kalman Filter used in the field of navigation is a minimum variance estimation that can be defined simply through the use of a conditional expectation [29]:

$$\delta \widehat{x}_{j/k} = E\left[\delta x_j \mid z_1, z_2, \ldots, z_k\right], \tag{16}$$

where $E[\cdot]$ is the expectation operator; $\delta x$ is the state vector; $z$ represents the measurements from time $t_1$ to $t_k$.

Considering $\phi_{k+1/k-1} = \phi_{k+1/k}\phi_{k/k-1}$, the state vector $\delta x_j$ can be deduced from formula (4) as

$$\delta x_j = \phi_{j/k}\delta x_k + \sum_{i=k+1}^{j} \phi_{j/i} G_{i/i-1} w_{i-1}, \quad k < j. \tag{17}$$

Combining (16) and (17) yields

$$\begin{aligned}
\delta \widehat{x}_{j/k} &= E\left[\delta x_j \mid z_1, z_2, \ldots, z_k\right] \\
&= E\left[\left(\phi_{j/k}\delta x_k + \sum_{i=k+1}^{j} \phi_{j/i} G_{i/i-1} w_{i-1}\right) \mid z_1, z_2, \right. \\
&\left. \ldots, z_k\right] = E\left[\phi_{j/k}\delta x_k \mid z_1, z_2, \ldots, z_k\right] \\
&+ \sum_{i=k+1}^{j} \phi_{j/i} G_{i/i-1} E\left[w_{i-1} \mid z_1, z_2, \ldots, z_k\right].
\end{aligned} \tag{18}$$

According to formulas (4) and (8), $w_{i-1}$ only has an effect on the state vector $\delta x_{i-1}$ and is irrelevant to the measurements $z_1, z_2, \ldots, z_k$. Additionally, $w_{i-1}$ is white noise, which is a sequence of zero-mean random variables that is uncorrelated timewise. Its expectation is given by $w_{i-1}$. Therefore, (18) can be simplified as follows:

$$\delta \widehat{x}_{j/k} = \phi_{j/k}\left[\delta x_k \mid z_1, z_2, \ldots, z_k\right] = \phi_{j/k}\delta \widehat{x}_k, \tag{19}$$

where $\delta \widehat{x}_k$ is the updating estimate of $\delta x_k$.

Defining $\delta \widetilde{x}_{j/k}$ as the error of $\delta x_j$ and $\delta \widehat{x}_{j/k}$,

$$\begin{aligned}
\delta \widetilde{x}_{j/k} &= \delta x_j - \delta \widehat{x}_{j/k} \\
&= \phi_{j/k}\delta x_k + \sum_{i=k+1}^{j} \phi_{j/i} G_{i/i-1} w_{i-1} - \phi_{j/k}\delta \widehat{x}_k \\
&= \phi_{j/k}\delta \widetilde{x}_k + \sum_{i=k+1}^{j} \phi_{j/i} G_{i/i-1} w_{i-1}.
\end{aligned} \tag{20}$$

The covariance matrix associated with $w_{i-1}$ is given by [23]

$$E\left[w_{i-1} {w_{i-1}}^T\right] = Q_{i-1}. \tag{21}$$

The covariance of $\delta \widetilde{x}_{j/k}$ can be

$$\begin{aligned}
P_{j/k}^- &= E\left[\delta \widetilde{x}_{j/k} {\delta \widetilde{x}_{j/k}}^T\right] \\
&= \phi_{j/k} E\left[\delta \widetilde{x}_k {\delta \widetilde{x}_k}^T\right] \phi_{j/k}^T \\
&\quad + \sum_{i=k+1}^{j} \phi_{j/i} G_{i/i-1} E\left[w_{i-1} w_{i-1}^T\right] G_{i/i-1}^T \phi_{j/i}^T \\
&= \phi_{j/k} \widehat{P}_k \phi_{j/k}^T + \sum_{i=k+1}^{j} \phi_{j/i} G_{i/i-1} Q_{i-1} G_{i/i-1}^T \phi_{j/i}^T.
\end{aligned} \tag{22}$$

$G_{i/i-1}$ is a positive-definite matrix, which satisfies $G_{i/i-1} Q_{i-1} G_{i/i-1}^T = Q_{i-1}$.

Defining $M_{j,k+1} = \sum_{i=k+1}^{j} \phi_{j/i} Q_{i-1} \phi_{j/i}^T$ yields

$$\begin{aligned}
M_{k+1,k+1} &= G_{k+1/k} Q_k G_{k+1/k}^T = Q_k, \\
M_{j+1,k+1} &= Q_j + \phi_{j+1/j} M_{j,k+1} \phi_{j+1/j}^T.
\end{aligned} \tag{23}$$

Combining the above equations yields

$$\begin{aligned}
\delta \widehat{x}_{j/k} &= \phi_{j/k}\delta \widehat{x}_k, \\
P_{j/k}^- &= \phi_{j/k} \widehat{P}_k \phi_{j/k}^T + M_{j,k+1}
\end{aligned} \tag{24}$$

which means that the error state estimate and its error covariance can be obtained by using the accumulated state transition matrixes. The LiDAR sampling time is treated as time $t_k$ and the LiDAR/IMU integration finish time is $t_j$. The state estimate and its covariance in $t_k$ can be propagated to $t_j$ by using formula (24), and then the propagated results can be updated using function (14). Finally, the updated state estimate is fed back to the current IMU output to correct the final output of navigation in the online EKF.

## 3. Results and Discussion

*3.1. System Overview.* To verify the online performance of our proposed integrated system, a series of tests based on the UGV mobile mapping platform has been designed. The hardware and software platform in this paper is what used

Table 1: The detailed parameters of LiDAR and IMU.

| LiDAR | | | IMU |
|---|---|---|---|
| Product model | Hokuyo UTM-EX | Product model | MEMS-level MTi-G |
| Sampling frequency | 10 Hz | Sampling frequency | 200 Hz |
| Scan range | 0.1 m–30 m | Gyroscope bias | 200 degrees/h |
| Scan angle | 270 degrees | Accelerometer bias | 2000 mGal |
| Angular resolution | 0.25 degrees | | (1 Gal = 1 cm/s$^2$) |

Table 2: The static positioning error statistics.

| | | RMS error | Mean error | Maximum error |
|---|---|---|---|---|
| Post-EKF | North | 0.0013 (m) | 0.0011 (m) | 0.0057 (m) |
| | East | 0.0036 (m) | 0.0030 (m) | 0.0132 (m) |
| | Heading | 0.2030 (degree) | 0.2019 (degree) | 0.4243 (degree) |
| Online EKF | North | 0.0033 (m) | 0.0017 (m) | 0.0127 (m) |
| | East | 0.0046 (m) | 0.0037 (m) | 0.0154 (m) |
| | Heading | 0.2319 (degree) | 0.2386 (degree) | 0.2869 (degree) |

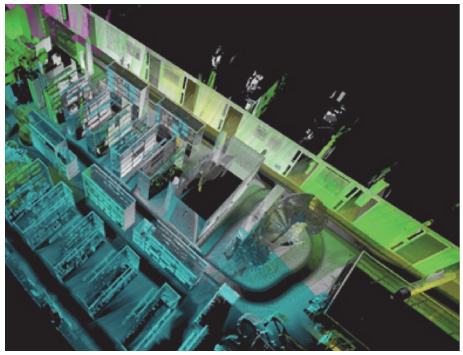


Figure 3: Top view of FGI library.

in previous work called NAVIS [5]. The detail information of LiDAR and IMU is listed in Table 1. The components are installed horizontally on the UGV platform. The whole system moves at a speed of about 0.9 m/s, and the total time spent on the experiments in this paper is approximately three minutes and forty-six seconds. The following data processing was conducted on a pad, with the Windows 8 operation system and a 1.6 GHz CPU, which is suitable for the online process for the UGV.

The field tests of the proposed LiDAR/IMU integrated system were conducted along the corridor of Finnish Geospatial Research Institute (FGI) library (see Figure 3). To evaluate the effectiveness of our proposed online EKF algorithm, stationary and dynamic experiments were carried out, and the mapping results generated by online EKF algorithm and postprocessed EKF algorithm were compared with that generated by a high precision laser scanner.

*3.2. Accuracy Evaluation of Stationary Estimation.* As shown in Figure 4, the stationary test was conducted from the beginning to 50th second. Table 2 shows the numerical positioning error statistic results of the two methods. Absolute position and the heading angle in the stationary moment were taken as reference. The overall position result in this paper has been projected in n-frame. The position RMS errors of north with postprocessed EKF method and online EKF method were 13 mm and 33 mm; the position RMS errors of east were 36 mm and 46 mm; the RMS errors of the heading were 0.2030 degrees and 0.2319 degrees, respectively. The difference in RMS of the two methods was approximately 2.0 mm, 1.0 mm, and 0.03 degrees. Considering that the range error of LiDAR is approximately 2–4 cm, and the angular resolution is about 0.25 degrees, such differences are acceptable. Thereby, in stationary positioning, the overall estimated accuracy of the position and attitude of the online EKF algorithm can be regarded as being in accordance with that of postprocessed EKF algorithm.

*3.3. Accuracy Evaluation of Dynamic Estimation.* Figure 5(a) is generated by LiDAR scan matching standalone. Figure 5(b) is the likelihood map generated with the online EKF algorithm. They are both the floor plan of the FGI library. By comparing Figures 5(a) and 5(b), it can be seen that the map results of LiDAR standalone system are much noisier than that of the LiDAR/INS, and there appeared mismatching in the long corridor in Figure 5(a), where the features are little. Therefore, the LiDAR/INS system can overcome the drawbacks of the LiDAR standalone system and achieve higher mapping accuracy.

The trajectories of the UGV platform in dynamic mode with postprocessed EKF algorithm and online algorithm are also shown in Figure 5(b). The initial position was taken as origin. As shown in the plot, the green trajectory coincides well with the red trajectory, which implies that our online EKF method has the same positioning accuracy as that of the sequential postprocessed EKF method. The difference is not obvious from the plot.

Table 3: The dynamic positioning difference statistics between online EKF and post-EKF.

| | RMS error | Mean error | Maximum error |
|---|---|---|---|
| North | 0.0138 (m) | 0.0115 (m) | 0.0379 (m) |
| East | 0.0440 (m) | 0.0343 (m) | 0.0983 (m) |
| Heading | 0.1979 (degree) | 0.1513 (degree) | 0.5422 (degree) |

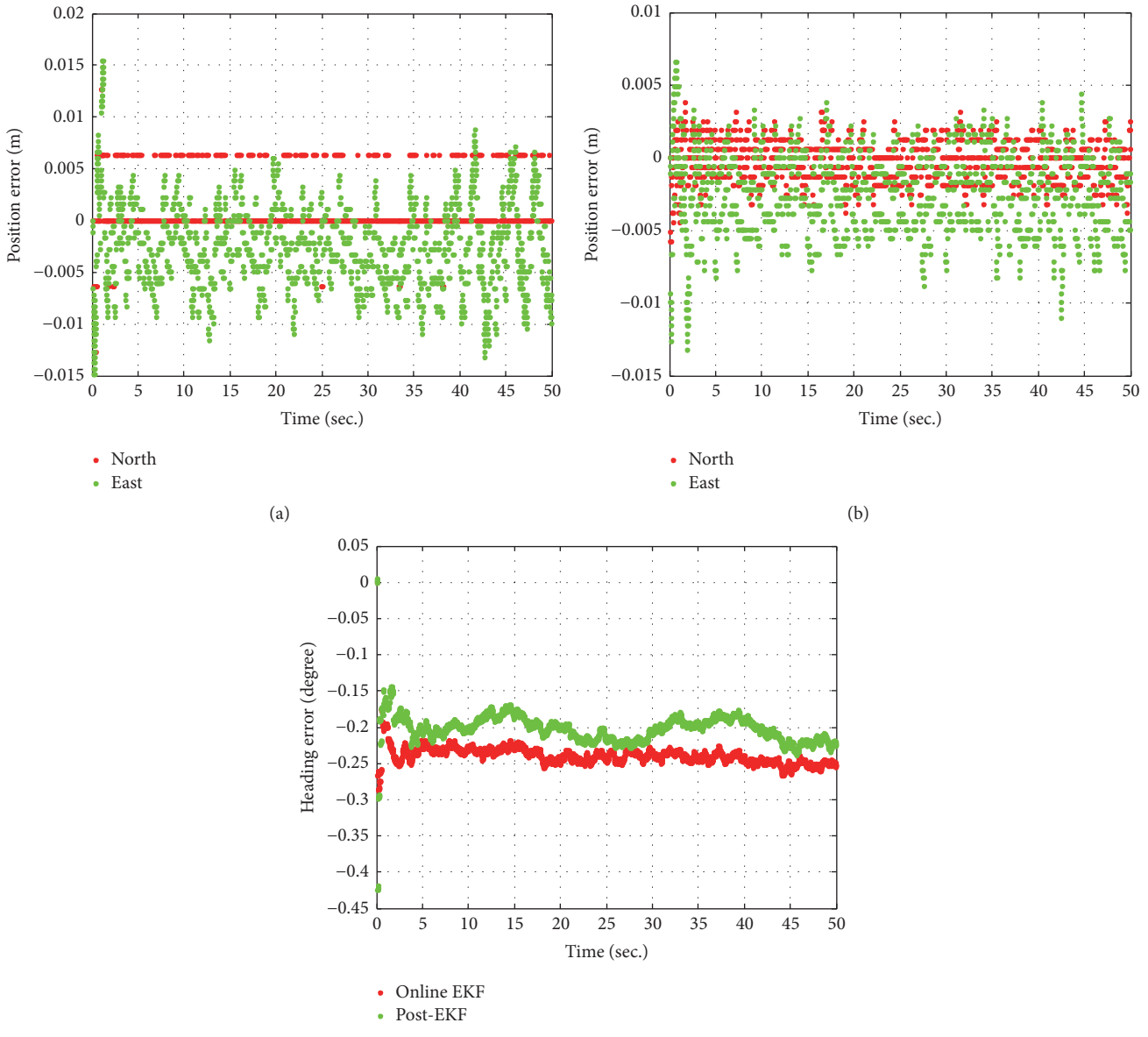



Figure 4: (a) The positioning results with online EKF solution; (b) the positioning results with post-EKF solution; (c) the heading results with online EKF and post-EKF.

The overall dynamic process lasted approximately 176 s. Figure 6 is the positioning difference with postprocessed EKF solution and the online EKF solution, and the statistics result is listed in Table 3. The position RMS errors of north and east are 0.0138 m and 0.0440 m, respectively; the RMS error of the heading is 0.1979 degrees. The position difference is still at centimeter-level, and the heading result is also under the angular resolution of LiDAR. As a result, the accuracy of the position and attitude of the online EKF algorithm can be considered as being at the same level with that of the postprocessed EKF in dynamic mode.

In addition to the comparison of trajectory accuracy, the map quality also needs to be verified. Figures 7(a)

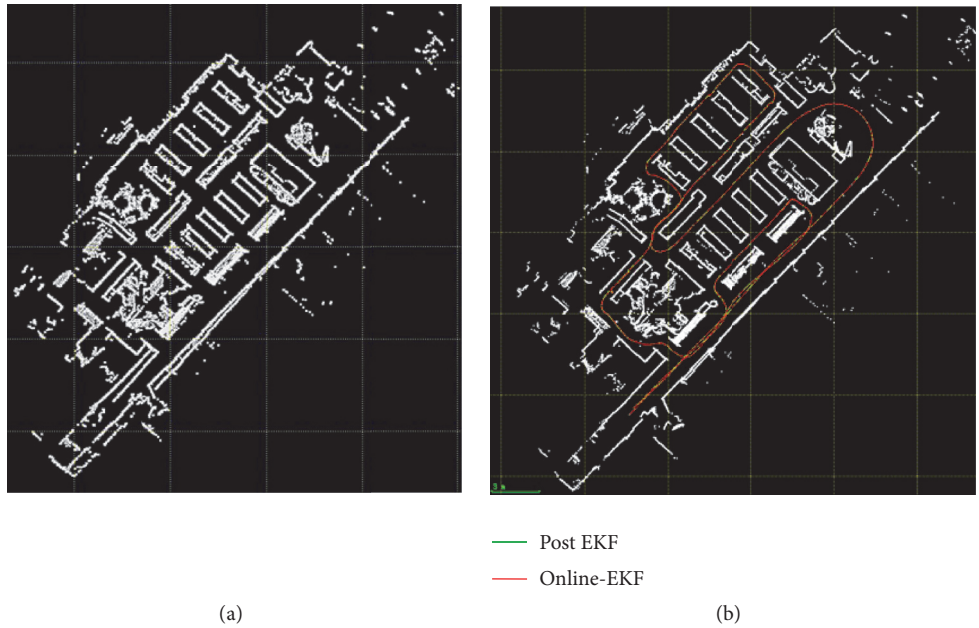



Figure 5: (a) The map results of UGV platform with LiDAR scan matching standalone. (b) The map and trajectories result of UGV platform with post-EKF solution and online EKF solution.

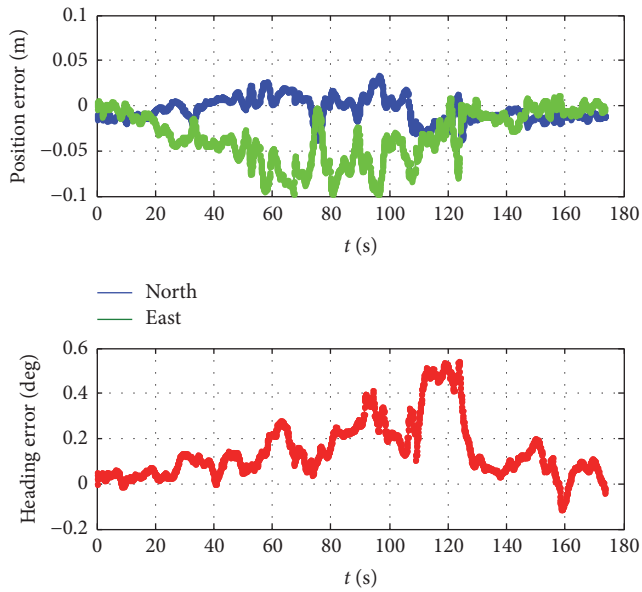



Figure 6: The comparison of positioning outputs by the post-EKF solution and online EKF solution.

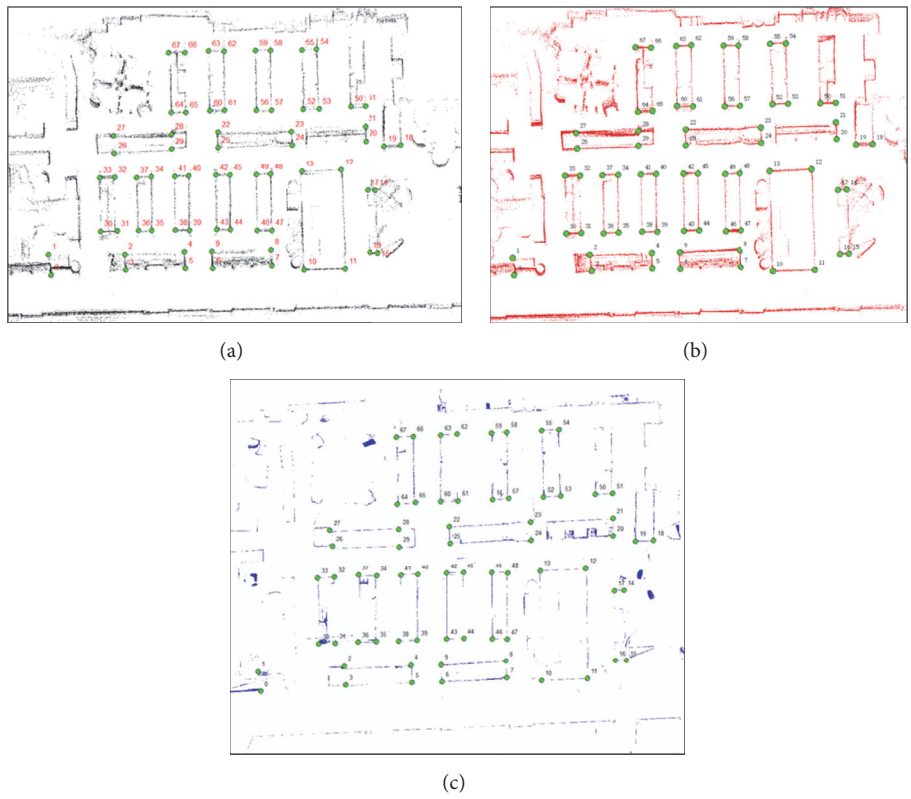
and 7(b) show the likelihood map generated by the two different solutions—postprocessed EKF solution and online EKF solution. They are compared with the reference map, which is presented in Figure 7(c) and generated with a Terrestrial Laser Scanner (TLS, *FARO Focus*$^{3D}$ *330X*). By comparing the zoom-in-images of each map, it is obvious that the line features in Figures 7(a) and 7(b) are remarkably noisier than that in Figure 7(c). The profiles are wider than their counterparts generated by TLS, and some of the corners are too ambiguous to be detected. The reason is that the adopted Hokuyo laser scanner is a big footprint scanner with centimeter ranging accuracy but the TLS applies small footprint millimeter accuracy laser.

Table 4: The comparison of accuracy results statistics for the selected feature points.

| | Post-EKF | Online EKF |
|---|---|---|
| Feature points | 67 | 67 |
| RMS (m) | 0.0562 | 0.0732 |

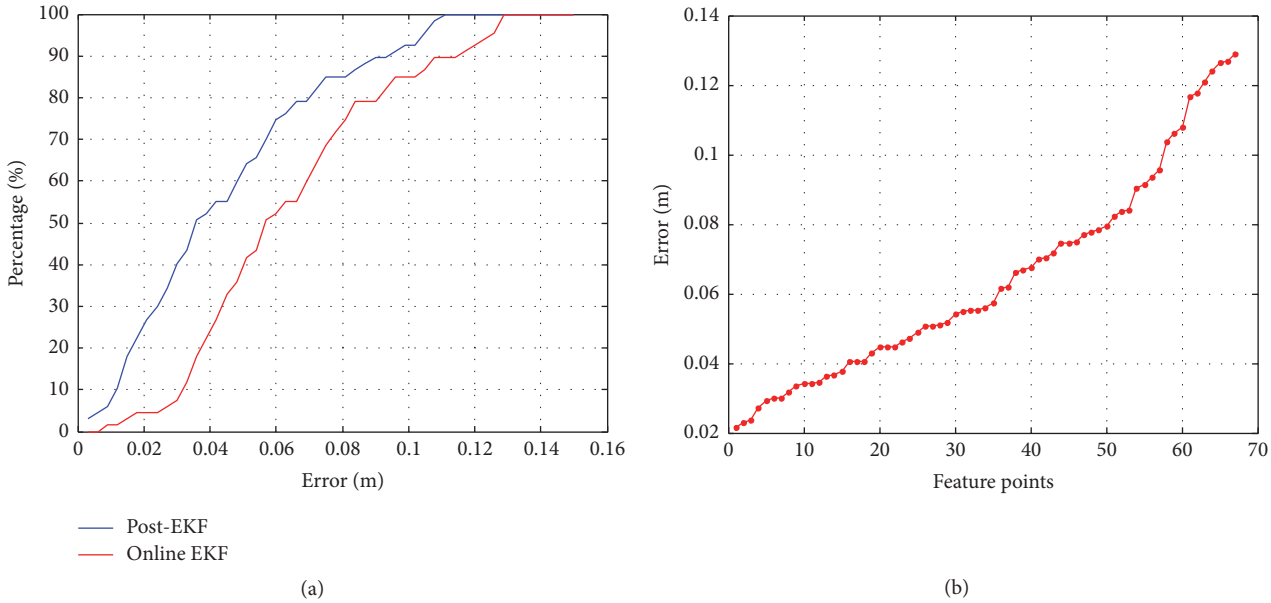
The unmovable corners of book shelves and walls are selected as the main feature points for accuracy evaluation. There are in total 67 feature points of corners picked out from the three maps for evaluation. The RMS errors of the selected feature points of the FGI library with the postprocessed EKF solution and online EKF solution are listed in Table 4. The RMS error with the postprocessed EKF method is 0.0562 m, and the RMS error with the online EKF method is 0.0732 m. The difference of RMS errors with the two methods is under 0.02 m. Considering the errors brought by manual operation, the accuracy is reasonable.

The positioning precision is reflected by the mapping precision indirectly in the indoor environment, due to the lacking of trajectory reference truth. Figure 8(a) shows the cumulative distribution of the mapping results errors with post-EKF and online EKF, respectively. The $y$-axis means the percentage of less than the corresponding errors in $x$-axis. The mapping results errors of feature points with post-EKF are all less than 0.011 m, and the errors of 70%–80% feature points are less than 0.06 m. While the errors of feature points with online EKF are all less than 0.013 m, the errors of 70%–80% feature points are less than 0.08 m. Figure 8(b) describes the drift of mapping results with online EKF. The



(a) (b) (c)

Figure 7: (a) NAVIS map result with selected feature points with online EKF solution; (b) NAVIS map result with selected feature points with post-EKF solution; (c) TLS reference map and the selected feature points.




(a) (b)

Figure 8: (a) The cumulative distribution of the mapping errors with post-EKF and online EKF; (b) the mapping error drift of the online EKF.

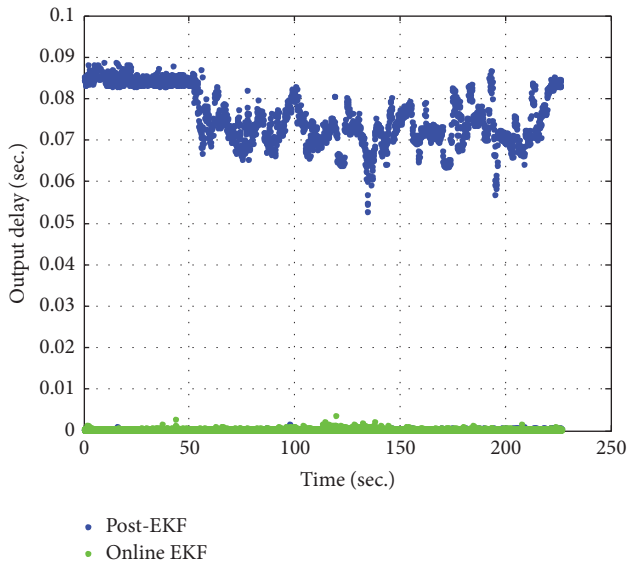



Figure 9: The output delay using online EKF solution and post-EKF solution.

errors of LiDAR and IMU will be accumulated continuously over time, if there is no loop closure or some other methods to correct the errors. The final drift is about 0.013 m.

*3.4. Verification of Real-Time Performance.* To evaluate the performance of online EKF solution, the time delay of each output epoch is compared with those of postprocessed EKF solution in Figure 9. Theoretically, as introduced in Figure 2, if there is no LiDAR data received, the IMU mechanization results will be output, and the output delay of one epoch means the time consumption of IMU mechanization. When the LiDAR data is received, for the traditional sequential EKF, the output delay of one epoch equals the total time it takes to complete the LiDAR scan matching, IMU mechanization, and Kalman update, but for the proposed parallel online EKF, the output delay of each epoch is composed of only the IMU mechanization, Kalman update, and the one-step-error-state-transition process, which will be much shorter.

As the results shown in Figure 9, the time delay of each output epoch with the online EKF is much less than that with the postprocessed EKF. As observed in Table 5 the mean output delay using the postprocessed EKF is 1.69 ms, while the maximum is 88.65 ms. The mean time delay of each output epoch using the online EKF is 0.280 ms, while the maximum is 3.76 ms. The mean output delay of the online EKF reduced about 6 times, and the maximum value reduces 23 times. The mean and maximum value improvement is because the online EKF can take advantage of the parallel processing, that is, dual processing threads running on two CPU cores, and the online EKF utilizes the one-step-error-state-transition method to make it possible to use previous update information (LiDAR scan matching) to correct current IMU mechanization results, so that IMU output does not have to wait until the LiDAR scan matching process finished. Therefore, according to Figure 9, the improved EKF method in this paper can reduce the output delay efficiently compared with the postprocessed EKF method, and it ensures the better real-time performance for the online system.

Table 5: The output delay statistics with online-EKF solution and post-EKF solution.

| | Mean | Maximum |
|---|---|---|
| Post-EKF | 1.69 ms | 88.65 ms |
| Online EKF | 0.28 ms | 3.76 ms |

## 4. Conclusions

An online solution for the LiDAR/IMU integrated system is proposed in this paper. The positioning results of IMU and LiDAR scan matching are real-time synchronized using the proposed one-step-error-state-transition method in the EKF to improve the real-time response of the LiDAR/IMU integrated navigation. The accuracy and online improvement results prove that (1) the proposed online method can achieve the same positioning and mapping accuracy (centimeters level) as the sequential post-EKF method; (2) the online EKF solution can reduce the maximum output delay from 88.65 ms in postprocessed EKF to 3.76 ms. It improves the real-time performance effectively. In conclusion, the online solution proposed in this paper can solve the time lag issue of LiDAR/IMU integration without the loss of the positioning accuracy. In the future work, the proposed method will be integrated into embedded-hardware platform and applied for real-time 2D and 3D indoor mapping.

## Conflicts of Interest

The authors declare no conflicts of interest.

## Acknowledgments

This study was financially supported by the National Key Research and Development Program (nos. 2016YFB0502202 and 2016YFB0501803) and the Academy of Finland (Project: "Centre of Excellence in Laser Scanning Research" (272195)). The authors express thanks to Yuwei Chen, Professor Juha Hyyppä, and other friends in Finnish Geospatial Research Institute, who provided the experiment data and useful feedback.

## References

[1] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Proceedings of the Robotics: Science and System Conference*, 2014.

[2] A. A. Aghamohammadi, H. D. Taghirad, A. H. Tamjidi et al., "Feature-based laser scan matching for accurate and high speed mobile robot localization," in *Proceedings of the European Conference on Mobile Robots EMCR '07*, Freiburg, Germany, September 2007.

[3] Z. Duan and Z. Cai, "Robust simultaneous localization and mapping based on laser range finder with improved adaptive

particle filter," in *Proceedings of the Chinese Control and Decision Conference CCDC '08*, pp. 2820–2824, July 2008.

[4] J. Tang, Y. Chen, A. Jaakkola, J. Liu, J. Hyyppä, and H. Hyyppä, "NAVIS-An UGV indoor positioning system using laser scan matching for large-area real-time applications," *Sensors*, vol. 14, no. 7, pp. 11805–11824, 2014.

[5] J. Tang, Y. Chen, X. Niu et al., "LiDAR scan matching aided inertial navigation system in GNSS-denied environments," *Sensors*, vol. 15, no. 7, pp. 16710–16728, 2015.

[6] I. Klein and S. Filin, "Lidar and INS fusion in periods of GPS outages for mobile laser scanning mapping systems," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 3812, pp. 231–236, 2011.

[7] A. Soloviev, "Tight coupling of GPS, laser scanner, and inertial measurements for navigation in urban environments," in *Proceedings of the IEEE/ION Position, Location and Navigation Symposium*, pp. 511–525, IEEE, Monterey, Calif, USA, May 2008.

[8] R. Li, J. Liu, L. Zhang, and Y. Hang, "LIDAR/MEMS IMU integrated navigation (SLAM) method for a small UAV in indoor environments," in *Proceedings of the 8th International Conference on DGON Inertial Sensors and Systems, ISS '14*, pp. 1–5, 2015.

[9] L. Chen, S. Ali-Löytty, R. Piché, and L. Wu, "Mobile tracking in mixed line-of-sight/non-line-of-sight conditions: algorithm and theoretical lower bound," *Wireless Personal Communications*, vol. 65, no. 4, pp. 753–771, 2012.

[10] N. Joshi, M. Baumann, A. Ehammer et al., "A review of the application of optical and radar remote sensing data fusion to land use mapping and monitoring," *Remote Sensing*, vol. 8, no. 1, article no. 70, 2016.

[11] L. Chen and L. Wu, "Mobile positioning in mixed LOS/NLOS conditions using modified EKF banks and data fusion method," *IEICE Transactions on Communications*, vol. 92, no. 4, pp. 1318–1325, 2009.

[12] H. B. Mitchell, "Multi-Sensor Data Fusion," in *Proceedings of the Intelligent Problem Solving, Methodologies and Approaches, International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE '00*, pp. 245–253, Louisiana, La, Usa, 2016.

[13] L. Chen, L. Pei, H. Kuusniemi, Y. Chen, T. Kröger, and R. Chen, "Bayesian fusion for indoor positioning using bluetooth fingerprints," *Wireless Personal Communications*, vol. 70, no. 4, pp. 1735–1745, 2013.

[14] Y. Chen, J. Liu, A. Jaakkola et al., "Knowledge-based indoor positioning based on LiDAR aided multiple sensors system for UGVs," in *Proceedings of the 2014 IEEE/ION Position, Location and Navigation Symposium, PLANS '14*, pp. 109–114, May 2014.

[15] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," in *Proceedings of the 9th IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR '11)*, pp. 155–160, IEEE, Kyoto, Japan, November 2011.

[16] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[17] R. Vincent, B. Limketkai, and M. Eriksen, "Comparison of indoor robot localization techniques in the absence of GPS," in *Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XV of Defense, Security, and Sensing Symposium*, Proceedings of the SPIE, April 2010.

[18] S. Thrun and J. J. Leonard, "Simultaneous localization and mapping," in *Springer Handbook of Robotics*, pp. 871–889, Springer Berlin Heidelberg, Berlin, Germany, 2008.

[19] R. Kaijaluoto, A. Kukko, and J. Hyyppä, "Precise indoor localization for mobile laser scanner," in *Proceedings of the ISPRS WG IV/7 and WG V/4 Joint Workshop on Indoor-Outdoor Seamless Modelling, Mapping and Navigation*, pp. 1–6, May 2015.

[20] J. Jung, S. Yoon, S. Ju, and J. Heo, "Development of kinematic 3D laser scanning system for indoor mapping and as-built BIM using constrained SLAM," *Sensors*, vol. 15, no. 10, pp. 26430–26456, 2015.

[21] J. Jung, S. Hong, S. Jeong et al., "Productive modeling for development of as-built BIM of existing indoor structures," *Automation in Construction*, vol. 42, no. 2, pp. 68–77, 2014.

[22] A. Fernández, P. F. Silva, I. Colomina et al., "Real-time navigation and mapping with mobile mapping systems using LiDAR/Camera/INS/GNSS advanced hybridization algorithms: description and test results," in *Proceedings of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS 2014*, pp. 896–903, September 2014.

[23] J. E. Guivant and E. M. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-time implementation," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 242–257, 2001.

[24] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte, "An efficient approach to the simultaneous localisation and mapping problem," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pp. 406–411, usa, May 2002.

[25] C. Estrada, J. Neira, and J. D. Tardós, "Hierarchical SLAM: real-time accurate mapping of large environments," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 588–596, 2005.

[26] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation, ICRA 2016*, pp. 1271–1278, swe, May 2016.

[27] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i the essential algorithms," *IEEE Robotics Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[28] Z. Hongping, L. Chang, W. Hongchen et al., "Improvement and verification of real-time performance of GNSS/INS tightly coupled integration in embedded platform," *Journal of Southeast University (Natural Science Edition)*, vol. 46, no. 4, pp. 695–701, 2016.

[29] E.-H. Shin and E.-S. Naser, *Accuracy Improvement of Low Cost INS/GPS for Land Applications*, University of Calgary, Department of Geomatics Engineering, Calgary, Canada, 2001, 2001.

[30] E. H. Shin, Estimation Techniques for Low-Cost Inertial Navigation UCGE report, 20219, 2005, http://www.geomatics.ucalgary.ca/links/GradTheses.html.

[31] X. Dong and S. Zhang, *Positioning and application of GPS/INS integrated navigation [M.S. thesis]*, National University of Defense Technology, Changsha, China, 1998.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 8260746, 16 pages
https://doi.org/10.1155/2017/8260746

*Research Article*

# Ubiquitous and Seamless Localization: Fusing GNSS Pseudoranges and WLAN Signal Strengths

**Philipp Richter and Manuel Toledano-Ayala**

*Universidad Autónoma de Querétaro, Santiago de Querétaro, QRO, Mexico*

Correspondence should be addressed to Philipp Richter; philipp.richter@uaq.mx

Received 10 December 2016; Revised 24 March 2017; Accepted 10 April 2017; Published 28 June 2017

Academic Editor: Olivier Julien



Ubiquitous global positioning is not feasible by GNSS alone, as it lacks accurate position fixes in dense urban centres and indoors. Hybrid positioning methods have been developed to aid GNSS in those environments. Fingerprinting localization in wireless local area networks (WLANs) is a promising aiding system because of its availability, accuracy, and error mechanisms opposed to that of GNSS. This article presents a low-cost approach to ubiquitous, seamless positioning based on a particle filter integrating GNSS pseudoranges and WLAN received signal strength indicators (RSSIs). To achieve accurate location estimates indoors/outdoors and in the transition zones, appropriate likelihood functions are essential as they determine the influence of each sensor information on the position estimate. We model the spatial RSSI distributions with Gaussian processes and use these models to predict RSSIs at the particle's positions to obtain point estimates of the RSSI likelihood function. The particle filter's performance is assessed with real data of two test trajectories in an environment challenging for GNSS and WLAN fingerprinting localization. Outcomes of an extended Kalman filter using pseudoranges and a WLAN position as observation is included as benchmark. The proposed algorithm achieves accurate and robust seamless localization with a median accuracy of five meters.

## 1. Introduction

The global navigation satellite system (GNSS) is present in almost all domains of modern life and it provides the base for many applications and services ranging from transportation and logistics, via surveying and mapping, to an uncountable number of leisure activities. With the development of the Internet of Things, the amount of location-aware services and the demanded accuracy will increase further. In densely constructed urban areas and indoors, GNSS has deficits. Shadowing and multipath phenomenon, which are often the largest error sources in these situations [1], inhibit robust, ubiquitous positioning. GNSS alone is unable to satisfy many location based services.

A common strategy to achieve robust ubiquitous positioning is to support GNSS with additional sensor information. Different technologies are available and many of them have been used in combination with GNSS. Positioning systems based upon wireless local area networks (WLAN) are emerging from the large amount of options [2, 3]. This is due to the global dissemination of WLAN infrastructure (limiting the costs of its exploitation) and WLAN enabled devices and the trade-off between range and potential positioning accuracy.

The approaches to using WLANs for localization are diverse. In this work, we resort to WLAN location fingerprinting for the reasons set forth as follows. WLAN location fingerprinting is based on the recognition of prerecorded signal strength (RSSI) readings. These RSSIs are ideally unique at different locations. This ideal is usually not achieved. Nonetheless, a heterogeneous signal strength distribution, achieved by diverse signal attenuations, facilitates the recognition of signal strength patterns. That applies in environments with many obstacles and thereby improves the localization accuracy. Hence, this method works best indoors and in urban areas. In contrast, GNSS is based upon the signal propagation delays between the satellites and the receiver, so called pseudoranges. Exact GNSS positioning demands the line-of-sight between receiver and satellites. Open outdoor environments meet this condition best; harsh

GNSS environments are dense urban centres and indoors. The operation principles of both localization technologies differ completely. They are well suited to complement each other, thus motivating the integration of pseudoranges and received signal strengths.

Several authors studied hybrid positioning systems based on GNSS and WLAN. The approaches differ in the level of integration and in the kind of information used from each of the systems. In a survey of these approaches, we found four categories in which the fusion of information of GNSS and WLAN can be grouped [4]: (1) WLAN assisted GNSS, (2) switching between position estimates of GNSS and of a WLAN based positioning system, (3) weighting the position estimates of GNSS and of a WLAN based system, and (4) deep integration of GNSS pseudoranges with different features of a WLAN based positioning system. We refer to [4] for the implications of the different integration schemes on the overall system and continue here with a brief overview of the works that study the fusion of GNSS and WLAN raw data, as they are the most relevant. They commonly rely on recursive Bayesian estimation methods, such as Kalman filter, particle filter, and their variants. Furthermore, all of them employ GNSS pseudoranges, but the features used from WLAN differ.

The authors of [5–9] deduce ranges from WLAN signals and combine them with pseudoranges. The WLAN ranges are obtained either from signal propagation times or from WLAN power measurements. In any case, the access point positions must be known, which is an unrealistic assumption in public areas, such as city centres. Another drawback of using ranges is the requirement of a line-of-sight between transmitter and receiver. This line-of-sight condition is rare indoors and makes accurate, seamless indoor/outdoor positioning very difficult. The advantage of this approach is that the same physical quantity is fused. This means that to fuse the ranges one can use the same, well-known methods as for GNSS positioning.

We consider the fusion of pseudoranges directly with RSSIs more promising, because of their contrary error mechanisms. In [10, 11], this approach was already investigated; likelihood functions from pseudorange and RSS measurements are established in both works and then fused within a particle filter. However, as fingerprints are spatially discrete distributed, the accuracy depends strongly on the fingerprint density. This is a clear disadvantage for medium- to large-area localization applications, where the laborious construction of fingerprint radio maps presents the biggest drawback and challenge.

This study extends the previous works and provides a solution that deals with the discrete fingerprints. We present a particle filter that integrates pseudoranges and RSSIs based on their likelihood functions, where a particle approximation of the signal strength likelihood function is obtained from a Gaussian process regression model. This method provides an automatic, well balanced weighting of the two sensor information sources and achieves accurate, robust, and smooth seamless positioning. We compare the algorithm with an extended Kalman filter that feeds WLAN position estimates as observation into a regular GPS extended Kalman filter.

## 2. Interpolating WLAN Signal Strength

To perform seamless location estimation in this study, we consider the problem of fusing WLAN RSSI with GNSS pseudoranges on a continuous state space. Fingerprinting techniques are based on spatial samples to construct the required radio map. Interpolating RSSI provides two principle advantages. First, it reduces the labor intensive construction of radio maps, because less fingerprints are needed to obtain a fingerprint database that is equally exact compared to without interpolation. Second and more importantly here, it provides the continuous model for the fingerprint database, facilitating the fusion of RSSI and pseudoranges.

This section explains interpolation of RSSI radio maps through Gaussian process regression.

*2.1. Preliminaries.* Gaussian processes are stochastic processes, basically a generalization of multivariate Gaussian distributions to infinite dimension. Every point of the infinite input domain has a Gaussian random variable associated. The defining property is that any finite collection of those random variables has a joint multivariate Gaussian distribution. Gaussian processes are completely characterized by a mean function $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and a covariance function $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]$:

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}), k\left(\mathbf{x}, \mathbf{x}'\right)\right), \tag{1}$$

where $\mathbf{x}$ is a vector from the index set.

*2.2. Gaussian Process RSSI Model.* Consider a nonlinear function $s_n = h(\mathbf{p}_n) + \varepsilon_n$ describing the relationship between positions in space $\mathbf{p}_n$ and the corresponding RSSIs, $s_n$. The term $h(\cdot)$ is the latent function we seek in order to predict/interpolate RSSIs and $\varepsilon_n$ represents i.i.d noise. In practice, one deals with a finite collection of input points. Constructing a fingerprint radio map with $M$ fingerprints corresponds to drawing a set of noisy RSSI samples $\mathbf{s} \equiv \{s_n\}_{n=1}^{M}$ at known positions $P \equiv \{\mathbf{p}_n\}_{n=1}^{M}$ from that unknown function. In the context of machine learning, this data is called *training data*.

Gaussian process modelling can be seen from a Bayesian point of view. The basic characteristics of the latent function is described by a Gaussian process a priori distribution and the function that we want to infer is modelled by a Gaussian process a posteriori distribution. To model the latent function appropriately, the mean and covariance function of the prior distribution have to be chosen and their parameters, so called hyperparameters, need to be found. (The hyperparameters can be learned from the training data, which is described in Section 2.3.2.) The likelihood function of the Gaussian process modelling reflects how likely are the RSSI samples in the light of the model.

The observed RSSIs are spatially correlated; neighbouring RSSIs are usually more strongly correlated than RSSIs that are distant. This relationship between the RSSIs is modelled by a covariance function. Covariance functions are specified by kernels. For two different input positions in space, $\mathbf{p}_p$

and $\mathbf{p}_q$, these functions are one at zero input distance, $\|\mathbf{p}_p - \mathbf{p}_q\|$, and decay with increasing distance. In [12], it has been shown that a constant mean function and a Matérn covariance function fit RSSI data very well. The constant mean function constitutes a single hyperparameter and is simply $[m(\mathbf{p}_p), m(\mathbf{p}_q)]^T = c$. As in [12], we use the Matérn kernel (with $\nu = 3/2$ [13]). To account for the i.i.d observation noise, the kernel is enhanced with a noise term:

$$\begin{aligned}&\operatorname{cov}\left(h\left(\mathbf{p}_p\right)+\varepsilon_p, h\left(\mathbf{p}_q\right)+\varepsilon_q\right)\\&\quad=\sigma_f^2\left(1+\frac{\sqrt{3}}{\ell}\left\|\mathbf{p}_p-\mathbf{p}_q\right\|\right)\exp\left(-\frac{\sqrt{3}}{\ell}\left\|\mathbf{p}_p-\mathbf{p}_q\right\|\right)\\&\qquad+\sigma_\varepsilon^2\delta\left(\mathbf{p}_p,\mathbf{p}_q\right).\end{aligned}\tag{2}$$

This covariance function has three hyperparameters: $\sigma_f^2$ is the signal variance, basically scaling the RSSIs; $\ell$ is the length scale, determining the variability of the underlying process; and $\sigma_\varepsilon^2$ is the noise variance, specifying the power of the noise. The constant mean, $c$, specifies the value to which the Gaussian process converges if training data are absent.

*2.3. Gaussian Process Regression.* The Gaussian process that describes a finite collection of observed, noisy RSSI samples is a multivariate Gaussian distribution $\mathbf{s} \sim \mathcal{N}(\mu(\mathbf{s}), \operatorname{cov}(\mathbf{s}))$, with the mean vector $\mu(\mathbf{s}) = \mathbf{m}(P)$ and the covariance matrix $\operatorname{cov}(\mathbf{s}) = K(P,P)+\sigma_\varepsilon^2 I$. Here, $K(P,P)$ is a matrix that contains the covariances, determined by the Matérn kernel, for all pairs of the position matrix $P$.

The described relationship between already observed RSSIs extends as well to not yet seen RSSIs, that is, to values we wish to predict. We denote those RSSIs with $\mathbf{s}^*$ and the corresponding positions with $P^*$ and call the collection of these two sets *test data*. Both sets of RSSIs, the training outputs and the test outputs, originate from the same physical phenomena and are thus jointly Gaussian distributed:

$$\begin{bmatrix}\mathbf{s}\\ \mathbf{s}^*\end{bmatrix}\sim\mathcal{N}\left(\begin{bmatrix}\mathbf{m}(P)\\ \mathbf{m}(P^*)\end{bmatrix},\begin{bmatrix}K(P,P)+\sigma_\epsilon^2 I & K(P,P^*)\\ K(P^*,P) & K(P^*,P^*)+\sigma_\epsilon^2 I\end{bmatrix}\right).\tag{3}$$

The covariance matrix $K(P,P^*) = K(P^*,P)^T$ contains the covariances evaluated at all pairs of training and test positions and $K(P^*,P^*)$ at all pairs of test positions, respectively.

By conditioning the joint Gaussian distribution (3) on the observations and inputs, the Gaussian process posterior distribution is obtained:

$$\mathbf{s}^* \mid P, \mathbf{s}, P^* \sim \mathcal{N}\left(\mu\left(\mathbf{s}^*\right), \operatorname{cov}\left(\mathbf{s}^*\right)\right).\tag{4}$$

Its mean and covariance functions are given with

$$\begin{aligned}\mu\left(\mathbf{s}^*\right)&\equiv\mathbb{E}\left[\mathbf{s}^*\mid P,\mathbf{s},P^*\right]\\&=\mathbf{m}\left(P^*\right)\\&\quad+K\left(P^*,P\right)\left[K\left(P,P\right)+\sigma_\varepsilon^2 I\right]^{-1}\left(\mathbf{s}-\mathbf{m}\left(P\right)\right),\end{aligned}\tag{5}$$

$$\begin{aligned}&\operatorname{cov}\left(\mathbf{s}^*\right)\\&\quad=\left[K\left(P^*,P^*\right)+\sigma_\epsilon^2 I\right]\\&\qquad-K\left(P^*,P\right)\left[K\left(P,P\right)+\sigma_\varepsilon^2 I\right]^{-1}K\left(P,P^*\right).\end{aligned}\tag{6}$$

Equation (5) enables the prediction of RSSIs at arbitrary positions $P^*$ and (6) enables computing the covariance at these positions. These covariances provide a useful measure of the uncertainty of the predictions.

*2.3.1. Nonzero Mean Function.* The term $\mathbf{m}(\cdot)$ is a deterministic mean function. Regression with a fixed mean function is simple. The fixed mean is subtracted from the observations, reverting the Gaussian process to a zero mean process. Then the standard Gaussian process regression procedure is applied, after which the mean function is added again. By replacing the fixed mean function with a set of basis functions, the mean function can be parametrized. Its hyperparameters, the parameters of the basis functions, can then be inferred from the data [13].

*2.3.2. Finding the Hyperparameters.* A final point we need to mention is how to obtain the hyperparameter of the mean and covariance function. The likelihood function of Gaussian process regression, $p(\mathbf{s} \mid P, \theta)$, is the key to that.

For notational convenience, we collect the hyperparameters in a vector $\theta = \{c, \sigma_f, \ell, \sigma_n\}$. The posterior distribution's dependence on the hyperparameter is expressed by

$$p\left(\mathbf{s}^*\mid P^*,\mathbf{s},P\right)=\int p\left(\mathbf{s}^*\mid P^*,\mathbf{s},P,\theta\right)p\left(\theta\mid\mathbf{s},P\right)\mathrm{d}\theta,\tag{7}$$

an intractable integral. Nonetheless, it can be approximated by using the most probable values of the hyperparameters $p(\mathbf{s}^* \mid P^*, \mathbf{s}, P) \approx p(\mathbf{s}^* \mid P^*, \mathbf{s}, P, \theta_{\mathrm{mp}})$ [14]. The most probable hyperparameters $\theta_{\mathrm{mp}}$ can be determined based on the posterior probability of $\theta$, $p(\theta \mid \mathbf{s}, P)$. Maximizing

$$\widehat{\theta}=\arg\max_\theta\frac{p\left(\mathbf{s}\mid P,\theta\right)p\left(\theta\right)}{p\left(\mathbf{s}\mid P\right)}\tag{8}$$

allows inferring the hyperparameters. This fraction reduces to the marginal likelihood function, as the denominator is independent of $\theta$ and $p(\theta)$ is modelled uniformly (usually no prior knowledge about the hyperparameters is available). In practice, the inference is done by minimizing the negative log-likelihood:

$$\widehat{\theta}=\arg\min_\theta\left(-\ln p\left(\mathbf{s}\mid P,\theta\right)\right).\tag{9}$$

Assuming a normally distributed noise process and prior distribution, an analytic expression for the marginal likelihood can be derived [13]:

$$\ln p(\mathbf{s} \mid P, \theta) = -\frac{N}{2}\ln 2\pi - \frac{1}{2}\ln\left|K + \sigma_\epsilon^2 I\right| - \frac{1}{2}\left(\mathbf{s}^T\left(K + \sigma_\epsilon^2 I\right)^{-1}\mathbf{s}\right). \quad (10)$$

The optimal hyperparameters are found by computing the partial derivatives of this function with respect to the hyperparameters and then applying a gradient search algorithm to find them.

## 3. Sequential Bayesian Estimation

The localization problem can be optimally described by the sequential Bayesian filter, a recursive framework that also enables fusing the sensory data. It propagates the location estimate over time based on the previous estimate of the mobile terminal's position and arriving observations.

It consists of two steps that are recursively executed. The process update predicts the current location based on the previous one and a model of the mobile terminal's motion. The process update is expressed by the Chapman-Kolmogorov equation [15]:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})\, d\mathbf{x}_{k-1}, \quad (11)$$

where $\mathbf{x}_k$ is the state vector at discrete time $k$. The time index $1 : k$ denotes a sequence of variables, for example, all observations, $\mathbf{z}_{1:k} \equiv \{z_i,\ i = 1, \ldots, k\}$, obtained up to time $k$. The motion of the mobile terminal is modelled by the state transition probability density $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$. The term $p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})$ is the prior probability density. It expresses information about the mobile terminal available before the current estimation step and represents the previous state of the object or some initial state. As soon as observations are available, the predicted state can be corrected. This is done in the measurement update step:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})}{\int p(\mathbf{z}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})\, d\mathbf{x}_k}, \quad (12)$$

where $p(\mathbf{z}_k \mid \mathbf{x}_k)$ is the likelihood function [15]. It encodes the information of the observations and is used to correct the prediction to yield the posterior density $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$. This recursion estimates the location of the mobile terminal optimally; however, these integrals do usually not have an analytic solution and must be approximated.

*3.1. Particle Filter.* One of the most versatile approximations to the recursive Bayesian estimation problem is particle filters. Particle filters rely on sampling from functions to approximate them. The versatility stems from their ability to approximate arbitrary functions. A detailed description of the generalized particle filter can be found in [15], that we follow in this subsection.

Complex integrals, as the ones that describe the sequential Bayesian estimator, can be approximated numerically if we can sample the involved functions. If the samples cover the function's domain well, sums replace complex integrals while the simplified operation on the samples provides a sufficiently accurate approximation. We denote the samples by $\{x^{(i)},\ i = 1, \ldots, N\}$.

Nevertheless, it is often not possible to sample a function, as it is unknown or too complex. This issue is circumvented by sampling a similar function, $q(x)$, which is proportional to the target function $p(x^{(i)}) = \tilde{\omega}^{(i)} q(x^{(i)})$. The term $q(x^{(i)})$ is the importance density. The introduced weights $\tilde{\omega}^{(i)}$ compensate the difference between the importance density and the target density. These weights need to be normalized to sum to unity, so that the functions are valid probability densities: $\omega^{(i)} = \tilde{\omega}^{(i)} / \sum_{j=1}^{N} \tilde{\omega}^{(j)}$. The idea of approximating arbitrary functions by a set of weighted particles $\{x^{(i)}, \omega^{(i)}\}_{i=1}^{N}$ is known as importance sampling. Consider the samples $\{x^{(i)},\ i = 1, \ldots, N\} \sim q(x^{(i)})$. Thus, we can approximate a density $p(x)$ by samples:

$$p(x) \approx \sum_{i=1}^{N} \omega^{(i)} \delta\left(x - x^{(i)}\right). \quad (13)$$

In the context of recursive state estimation, $\mathbf{x}^{(i)}$ is the state of the $i$th particle and $\omega^{(i)}$ is its associated weight. The target density is the posterior density $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$. To this point, it was left open how the weights are obtained. They are first defined in general by

$$\omega_k^{(i)} \propto \frac{p\left(\mathbf{x}_{0:k}^{(i)} \mid \mathbf{z}_{1:k}\right)}{q\left(\mathbf{x}_{0:k}^{(i)} \mid \mathbf{z}_{1:k}\right)}. \quad (14)$$

The derivation of the weights is based on a factorization of the importance density. A factorization that allows augmenting the importance density of the previous time step with a factor that represents the current state, so that particles that are distributed according to the importance density of the previous state can be updated with particles that approximate the current state. This leads to a recursive weight update which is given by

$$\omega_k^{(i)} \propto \omega_{k-1}^{(i)} \frac{p\left(\mathbf{z}_k \mid \mathbf{x}_k^{(i)}\right) p\left(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)}\right)}{q\left(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k\right)}. \quad (15)$$

Equation (15) allows approximating the posterior density by particles and their weights.

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} \omega_k^{(i)} \delta\left(\mathbf{x}_k - \mathbf{x}_k^{(i)}\right). \quad (16)$$

The weights must be normalized after each iteration.

The described procedure constitutes the problem of particle degeneration; that is, after each of the iterations, only a few particles gain weights while, for the majority of particles, the weight is reduced, until just one particle concentrates almost

the complete weight and the weight of the remaining particles is negligible.

The choice of the importance density may slow down this effect, but usually an additional resampling (with replacement) step is introduced to overcome it. During the resampling step, particles are multiplied according to their weights, so that particles that possess large weights are selected several times and particles with very small weights are selected once or possibly vanish.

As the resampling introduces further problems (principally reduces the diversity, because many particles are repeated), we resample when the degeneracy of the algorithm becomes severe. This can be assessed by the effective sample size $N_{\text{eff}} = 1/\sum_{i-1}^{N}(\omega_k^{(i)})^2$. We resample if the effective samples size exceeds 2/3 of the numbers of particles.

In this work, we choose the importance density to be the state transition density $q(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k) = p(\mathbf{x}_k^{(i)} \mid \mathbf{x}_{k-1}^{(i)})$. This simplifies (15) further; it reduces the weight update to be a product of the previous weights and the likelihood function:

$$\omega_k^{(i)} \propto \omega_{k-1}^{(i)} p\left(\mathbf{z}_k \mid \mathbf{x}_k^{(i)}\right). \tag{17}$$

## 4. Fusing GNSS Pseudoranges and WLAN Received Signal Strength

This section starts with an explanation of the used models in Section 4.1 and describes their integration into the Bayesian filter framework in Section 4.2. The extended Kalman filter that is used as benchmark is detailed at the end of this section.

*4.1. Models.* The state space for the self-localization problem is defined in a local level coordinate frame, where a position in space is denoted by $\mathbf{p} = (\mathsf{x}, \mathsf{y}, \mathsf{z})^T$. The corresponding velocity is its time derivative $\dot{\mathbf{p}} = (\dot{\mathsf{x}}, \dot{\mathsf{y}}, \dot{\mathsf{z}})^T$. We denote the state vector at time $k$ by $\mathbf{x}_k = (\mathbf{p}_k, \dot{\mathbf{p}}_k)^T$.

*4.1.1. Process Model.* To model the motion of the mobile terminal, we have chosen a random walk velocity process as in [16]. It stems from a Langevin process, a stochastic differential equation, that describes the velocity of a Brownian particle. For motion in a single dimension, this process reads

$$\begin{aligned} \dot{\mathsf{x}}_k &= a_{\mathsf{x}} \dot{\mathsf{x}}_{k-1} + b_{\mathsf{x}} v_{\mathsf{x}_k}, \\ \mathsf{x}_k &= \mathsf{x}_{k-1} + \Delta T \dot{\mathsf{x}}_k \end{aligned} \tag{18}$$

with

$$\begin{aligned} a_{\mathsf{x}} &= \exp\left(-\beta_{\mathsf{x}} \Delta T\right), \\ b_{\mathsf{x}} &= \overline{v}_{\mathsf{x}} \sqrt{1 - a_{\mathsf{x}}^2}. \end{aligned} \tag{19}$$

It is driven by a white and normally distributed excitation process $v_{\mathsf{x}_k} \sim \mathcal{N}(0, 1)$, which allows the uncertainty of the velocity to grow during prediction and also enhances the diversity of the particles. Parameter $\beta_{\mathsf{x}}$ is the process' rate constant. It determines the velocity increment of the mobile terminal during a discretization time step $\Delta T$, where steady-state root-mean-square velocity, $\overline{v}_{\mathsf{x}}$, confines the velocity of the mobile terminal. For the experiments described above, we determined these parameters empirically to $\beta_{\mathsf{x},\mathsf{y}} = 10\,\mathrm{s}^{-1}$, $\beta_{\mathsf{z}} = 100\,\mathrm{s}^{-1}$ and $\overline{v}_{\mathsf{x},\mathsf{y}} = 1.5\,\mathrm{m\,s}^{-1}$, and $\overline{v}_{\mathsf{z}} = 0.5\,\mathrm{m\,s}^{-1}$.

This model results in a linear process model:

$$\mathbf{x}_k = F_{pv} \mathbf{x}_{k-1} + G_{pv} \mathbf{v}_{k-1}, \tag{20}$$

with the state transition and noise coupling matrix

$$F_{pv} = \begin{pmatrix} 1 & 0 & 0 & a_{\mathsf{x}}\Delta T & 0 & 0 \\ 0 & 1 & 0 & 0 & a_{\mathsf{y}}\Delta T & 0 \\ 0 & 0 & 1 & 0 & 0 & a_{\mathsf{z}}\Delta T \\ & & & a_{\mathsf{x}} & 0 & 0 \\ & \mathbf{0}_{3\times 3} & & 0 & a_{\mathsf{y}} & 0 \\ & & & 0 & 0 & a_{\mathsf{z}} \end{pmatrix}, \tag{21}$$

$$G_{pv} = \begin{pmatrix} b_{\mathsf{x}} & 0 & 0 \\ 0 & b_{\mathsf{y}} & 0 \\ 0 & 0 & b_{\mathsf{z}} \\ b_{\mathsf{x}}\Delta T & 0 & 0 \\ 0 & b_{\mathsf{y}}\Delta T & 0 \\ 0 & 0 & b_{\mathsf{z}}\Delta T \end{pmatrix}.$$

The noise covariance matrix is simply $V \equiv \mathbb{E}[\mathbf{v}_k, \mathbf{v}_i] = I$; thus, the state transition probability density is Gaussian with the corresponding mean and covariance matrix:

$$\mathbf{x}_k \mid \mathbf{x}_{k-1} \sim \mathcal{N}\left(F_{pv}\mathbf{x}_{k-1}, G_{pv} V G_{pv}^T\right). \tag{22}$$

This process model is rather unspecific and therefore very general. Matrix $F_{pv}$ implies basically a static mobile terminal; no specific motion or direction is presumed. Instead it increases the variance over time, thus broadening the region of the potential location. The choice of parameters increases the variance in east and north equally but restricts it in $\mathsf{z}$-direction. The reason for that setting is attributed to the lack of an useful geographic height in the radio map. As the $\mathsf{z}$-coordinate cannot be corrected in GPS denied areas, we basically slow down the motion in $\mathsf{z}$-direction considerably.

*4.1.2. GNSS Pseudorange Likelihood Function.* A pseudorange describes the distance from the satellite to the receiver. This consists of the geometric distance, the difference between the satellite clocks and the receiver clock, and an ionospheric and tropospheric correction term. Most of these terms can be corrected using information transmitted by the satellites. However, errors occurring in the user segment remain. This is the receiver clock offset, $\Delta t_k$, which can usually be estimated. Other considerable errors are attributed to multipath propagation and shadowing effects. We collect these in $w_{\eta,k}$, together with errors from the ephemerides

prediction, relativistic effects, residuals from the correction terms, and noise. The pseudorange of the $j$th satellite becomes

$$\rho_k^j = \left\| \mathbf{p}_{s,k}^j - \mathbf{p}_k \right\| + c\Delta t_k + w_{\eta,k}^j, \quad j = 1, \ldots, J. \tag{23}$$

The error term $w_{\eta,k}$ is commonly modelled as a normally distributed, zero mean random variable $w_{\eta,k} \sim \mathcal{N}(0, \sigma_{w_{\eta,k}}^2)$. Assuming furthermore that the pseudoranges from different satellites are independent, the joint pseudorange likelihood function for the collection of pseudoranges, $\boldsymbol{\rho}_k = \{\overline{\rho}_k^j\}_{j=1}^J$, can be written as

$$p\left(\boldsymbol{\rho}_k \mid \mathbf{x}_k = \mathbf{x}_q\right) = \prod_{j=1}^{J} \frac{1}{\sqrt{2\pi}\sigma_{w_\eta,k}^j} \exp\left( -\frac{1}{2} \frac{\left(\overline{\rho}_k^j - \rho_k^j\right)^2}{\left(\sigma_{w_\eta,k}^j\right)^2} \right). \tag{24}$$

It expresses the likelihood that the observed pseudoranges were measured at $\mathbf{p}_q = C\mathbf{x}_q \in \mathbb{R}^3$, where $C = (I_3 \mid \mathbf{0}_3)$. The standard deviation $\sigma_{w_\eta,k}^j$ is composed by various terms that are set or estimated by RTKLIB [17]; among these terms are a deviation estimate based on the user accuracy index and fix values set by RTKLIB accounting for the errors in estimating delays attributed to the troposphere, ionosphere, and code bias.

Notice that the receiver clock offset is not part of the state vector and it is not estimated by the particle filter. As in [11], the least squares method is used iteratively to estimate the receiver clock offset.

*4.1.3. WLAN RSSI Likelihood Function.* A likelihood function for the RSSI observations, $s_k$, is established as follows. As we work with time averaged RSSI, to compensate the variations and to get closer to the assumption of normally distributed observations, we denote the arithmetic mean of RSSIs by $\overline{s}_k$. To evaluate the likelihood of an observation at different positions, a model predicting the observation, $\widehat{s}_k$, is required. However, the lack of an accurate model that relates RSSI and space is the reason to resort to the empirical method of fingerprinting. As RSSI measurements only exist at fingerprint locations, one has to approximate the RSSI measurement at the predicted position of the mobile terminal.

These RSSI estimates are obtained through Gaussian process regression; recall Section 2.3. Assume we have $l = 1, \ldots, L$ access points and that we receive their packets during survey and localization phase. The Gaussian process model for each access point can be trained off-line, before the actual positioning phase. Let $\overline{s}_k^l$ be the average of RSSIs received from the $l$th access point. Once the Gaussian process models are constructed, one uses (5) and (6) to predict a vector of RSSIs, $\mu(\mathbf{s}_k^{*,l})$, and the corresponding covariance matrix, $\mathrm{cov}(\mathbf{s}_k^{*,l})$, at the positions of interest $P_k^*$.

Finally, to compute the likelihood that $S_k = \{\overline{s}_k^l\}_{l=1}^L$ were observed at the position $\mathbf{p}_q^* \in P^*$, we establish a Gaussian likelihood function:

$$p\left(S_k \mid \mathbf{x}_k = \mathbf{x}_q\right) = \prod_{l=1}^{L} \frac{1}{\sqrt{2\pi}\sigma_{\mathbf{s}^*}^l} \exp\left( -\frac{1}{2} \frac{\left(\overline{s}_k^l - \mu_{\mathbf{s}^*}^l\right)^2}{\left(\sigma_{\mathbf{s}^*}^l\right)^2} \right), \tag{25}$$

where $\mu_{\mathbf{s}^*}^l$ is the interpolated RSSI and $\sigma_{\mathbf{s}^*}^l$ the corresponding standard deviation at the test input $\mathbf{p}_k^*$. This likelihood function is based on the assumption that the RSSI observations are independent. Notice, the radio map contains only two-dimensional location information; therefore $\mathbf{p}_q = D\mathbf{x}_q \in \mathbb{R}^2$ and the Boolean matrix is $D = (I_2 \mid \mathbf{0}_{2\times 4})$.

Compared with the pseudorange likelihood function, the predicted RSSI and variance are independent of time, because they are inferred from the (static) radio map. The same test position yields the same prediction. In contrast, the standard deviation in the pseudorange likelihood function is independent of the location of the mobile platform; it is only determined by the signal reception conditions. The choice of the RSSI variance arises from measurement setup. For many access points, only a single packet was captured; hence, a reasonable deviation measure could not be estimated. An alternative would be to exclude the observations from these access points and then use only the sample deviation measure from the remaining RSSI.

*4.2. Particle Filter Tight Fusing of Pseudoranges and Signal Strength.* This section combines the ideas previously described; it details the fusion of GPS pseudoranges and WLAN RSSI within the particle filter. Briefly, the key probability density functions of the sequential Bayesian estimator are approximated with particles and their weights and then plugged into the particle filter of Section 3.1.

*4.2.1. Process Update.* The probability density function that models the motion is already stated in (22). One could sample that density, compute the particle approximation to (11), and predict the mobile terminal's new location. Since an analytic process model is available, it is easier to propagate the particles that approximate the prior density, $p(\mathbf{x}_{k-1}^{(i)} \mid \mathbf{z}_{1:k-1})$, through the process model (20):

$$\mathbf{x}_k^{(i)} \mid \mathbf{z}_{1:k-1} = F_{pv}\mathbf{x}_{k-1}^{(i)} + G_{pv}\mathbf{v}_{k-1}^{(i)}, \tag{26}$$

where $\{\mathbf{v}_{k-1}^{(i)}\}_{i=1}^N$ are samples from the noise model.

When the filter is initialized, the prior density is usually not known. A common choice for $p(\mathbf{x}_0) \equiv p(\mathbf{x}_0 \mid \mathbf{z}_{-1})$ is a uniform density over a plausible area.

*4.2.2. Measurement Update.* The measurement update is formally defined by (12). The particle filter computes the posterior density through (16) and (17). At this step, a particle approximation of the predictive density is already available $p(\mathbf{x}_k^{(i)} \mid \mathbf{z}_{1:k-1})$, as are the weights from the previous time step.

Initially and after resampling, the weights are all equal and sum up to unity. What is left is to update the weights via the likelihood function.

A particle approximation of the pseudorange likelihood is simply obtained by computing the likelihood for each particle:

$$p\left(\boldsymbol{\rho}_k \mid \mathbf{x}_k = \mathbf{x}_{k_n}^{(i)}\right) = \prod_{j=1}^{J} \frac{1}{\sqrt{2\pi}\sigma_{w_\eta,k}^{j}} \exp\left(-\frac{1}{2}\frac{\left(\overline{\rho}_k^j - \rho_k^{j,(i)}\right)^2}{\left(\sigma_{w_\eta,k}^{j}\right)^2}\right), \tag{27}$$

where $\rho_k^{j,(i)}$ is a prediction of the $j$th pseudorange for a particle position according to the pseudorange model.

We proceed similarly for the RSSI likelihood function and compute the likelihood for each particle. This involves predicting a RSSI and variance value at each position of a particle; the RSSI likelihood function for the test positions $P_k^* = \{D\mathbf{x}_{k_n}^{(i)}\}_{i=1}^{N}$ becomes

$$p\left(S_k \mid \mathbf{x}_k = \mathbf{x}_{k_n}^{(i)}\right) = \prod_{l=1}^{L} \frac{1}{\sqrt{2\pi}\sigma_{\mathbf{s}^*,k}^{l,(i)}} \exp\left(-\frac{1}{2}\frac{\left(\overline{s}_k^l - \mu_{\mathbf{s}^*,k}^{l,(i)}\right)^2}{\left(\sigma_{\mathbf{s}^*,k}^{l,(i)}\right)^2}\right). \tag{28}$$

The term $\mu_{\mathbf{s}^*,k}^{l,(i)}$ denotes the RSSI predictions at the particle's positions and $\sigma_{\mathbf{s}^*,k}^{l,(i)}$ is the standard deviation at these positions, respectively.

Pseudoranges and RSSIs are clearly independent of each other, thus fusing RSSIs and pseudoranges equals multiplying their likelihood functions. An approximation of the posterior density is given by

$$p\left(\mathbf{x}_k \mid \mathbf{z}_{1:k}\right) \approx \sum_{i=1}^{N} \omega_{k-1}^{(i)} p\left(S_k \mid \mathbf{x}_k^{(i)}\right) p\left(\boldsymbol{\rho}_k \mid \mathbf{x}_k^{(i)}\right) \delta\left(\mathbf{x}_k - \mathbf{x}_k^{(i)}\right). \tag{29}$$

Likelihood functions provide information on the parameters only up to a multiplicative constant. To yield a valid probability density, the updated weights have to be normalized so that they sum up to unity.

In situations when no observation from one of the sensors was obtained, the likelihood function cannot be computed and is therefore ignored. If no measurement at all is available, the algorithm proceeds directly to the process update.

The influence of the measurements, either a pseudorange or a RSSI average, on the particles is controlled by these likelihood functions. On the one hand, the variances affect the particles weights. In the case of pseudoranges, it increases if the GPS observations are believed to be inexact or noisy. The variance of the RSSI likelihood function increases if the mobile terminal is in an area with few fingerprints, because, in these areas, information about RSSI is little. Large variances lead to small weights and vice versa. On the other hand, the position of a particle also affects its weight, because a particle far from the location indicated by an observation causes a large difference in the exponential of the functions, which decreases the weight. Due to these mechanisms, the proposed particle filter balances automatically between the two positioning systems.

*4.2.3. Position Estimate.* The final position estimate results from the minimization of the mean squared error, $\widehat{\mathbf{x}} = \arg\min_{\widehat{\mathbf{x}}} \mathbb{E}[(\widehat{\mathbf{x}} - \mathbf{x})^2 \mid \mathbf{z}]$ [18], and is known as the expected a posteriori estimate:

$$\widehat{\mathbf{x}}_k = \mathbb{E}\left[\mathbf{x}_k \mid \mathbf{z}_k\right] \equiv \int \mathbf{x}_k p\left(\mathbf{x}_k \mid \mathbf{z}_k\right) \mathrm{d}\mathbf{x}_k, \tag{30}$$

for the particle filter that becomes $\widehat{\mathbf{x}}_k = \sum_{i=1}^{N} \omega_k^{(i)} \mathbf{x}_k^{(i)}$.

We compute also the standard deviation of the particles as quality measure of the point estimate:

$$\sigma_p = \sqrt{\mathbb{E}\left[\left(\widehat{\mathbf{x}}_k - \mathbf{x}_k^{(i)}\right)^2\right]} = \sqrt{\frac{\sum_{i=1}^{N} \omega_k^{(i)} \left(\widehat{\mathbf{x}}_k - \mathbf{x}_k^{(i)}\right)^2}{\left(N' - 1\right)\sum_{i=1}^{N}\left(\omega_k^{(i)}\right)/N'}}, \tag{31}$$

where $N'$ denotes the nonzero weights [19].

*4.3. Benchmark Extended Kalman Filter.* To put the results of the proposed particle filter into context, we present them along with results obtained from an extended Kalman filter. This filter is based upon the process and measurement update stated in (11) and (12), but instead of propagating and correcting the probability densities it assumes the noises to be zero mean and normally distributed and therefore it propagates only the mean, $\widehat{\mathbf{x}}$, and the error covariance matrix, $P$, in time. Furthermore, we extend the state vector by the receiver clock offset, $\Delta t_k$, and its time derivative $\Delta \dot{t}_k$: $\mathbf{x}_k = (\mathbf{p}_k, \dot{\mathbf{p}}_k, \Delta t_k, \Delta \dot{t}_k)^T$.

We present first the process and measurement models and then the update and the correction step of the extended Kalman filter.

*4.3.1. Process Model.* The motion of the mobile terminal is described by the linear model (20). The process model of the receiver clock is according to [20] a two-state random process model

$$\begin{pmatrix} \Delta t_k \\ \Delta \dot{t}_k \end{pmatrix} = F_t \begin{pmatrix} \Delta t_{k-1} \\ \Delta \dot{t}_{k-1} \end{pmatrix} + G_t \begin{pmatrix} \nu_{1_{k-1}} \\ \nu_{2_{k-1}} \end{pmatrix}, \tag{32}$$

where the state transition matrix reads

$$F_t = \begin{pmatrix} 1 & \Delta T \\ 0 & 0 \end{pmatrix}. \tag{33}$$

The noise coupling matrix, $G_t$, can be obtained from the covariance matrix of the clock offset noise by $G_t = W\sqrt{D}$, where $W$ denotes the eigenvectors and $D$ the eigenvalues of $Q_t$; $\sqrt{\cdot}$ stands for the square root of the matrix' elements. The elements of the noise covariance matrix itself are determined

by the characteristics of the clock, given through the Allan variance coefficients $h_0$, $h_{-1}$, and $h_{-2}$:

$$\begin{aligned} q_{11} &= \frac{h_0}{2}\Delta T + 2h_{-1}\Delta T^2 + \frac{2}{3}\pi^2 h_{-2}\Delta T^3, \\ q_{12} &= h_{-1}\Delta T + \pi^2 h_{-2}\Delta^2, \\ q_{21} &= q_{12}, \\ q_{22} &= \frac{h_0}{2\Delta T} + 4h_{-1} + \frac{8}{3}\pi^2 h_{-2}\Delta, \end{aligned} \tag{34}$$

which can be found in [20] (We assume the GPS receiver clock to have a temperature compensated crystal clock.).

Combining the state transition and the noise coupling matrices of the motion and the receiver clock model

$$\begin{aligned} F &= \left(\begin{array}{c|c} F_{pv} & \mathbf{0}_{6\times 2} \\ \hline \mathbf{0}_{2\times 6} & F_t \end{array}\right), \\ G &= \left(\begin{array}{c} G_{pv} \\ \hline G_t \mid \mathbf{0}_{2\times 1} \end{array}\right), \end{aligned} \tag{35}$$

we can write the complete process model:

$$\mathbf{x}_k = F\mathbf{x}_{k-1} + G\mathbf{v}_{k-1}. \tag{36}$$

*4.3.2. Measurement Model.* Measurements to the extended Kalman filter are the pseudoranges and, if available, a position from WLAN. Thus, the measurement vector is $\mathbf{z}_k = (\rho_k^1, \ldots, \rho_k^J, \mathbf{p}_{\text{WLAN}})^T$. The measurement model is the common GPS measurement model for an eight-state extended Kalman filter; see [20]. Recalling the pseudorange model (23), we will abbreviate it here with $h(\cdot)$. Its Jacobian matrix reads

$$H_\rho = \begin{pmatrix} h_{\mathsf{x}}^1 & h_{\mathsf{y}}^1 & h_{\mathsf{z}}^1 & 0 & 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{\mathsf{x}}^J & h_{\mathsf{y}}^J & h_{\mathsf{z}}^J & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \tag{37}$$

where $h_{\mathsf{x}}^j$ stands for the partial derivative of the pseudorange measurement model (23) with respect to $\mathsf{x}$ for the $j$th satellite.

In the case that a position estimate from WLAN is available, the measurement matrix becomes simply

$$H = \left(\begin{array}{c} H_\rho \\ \hline I_{3\times 3} \mid \mathbf{0}_{3\times 5} \end{array}\right). \tag{38}$$

The position estimated from WLAN is a maximum likelihood estimate obtained from (28):

$$\widehat{\mathbf{x}}_{\text{WLAN}} = \arg\max_{\mathbf{x}} p\left(S_k \mid \mathbf{x}_k\right), \tag{39}$$

where $\widehat{\mathbf{p}}_{\text{WLAN}}$ corresponds to the first three elements of $\widehat{\mathbf{x}}_{\text{WLAN}}$.

For completeness, we denote the general measurement model including all observed pseudoranges and a position from WLAN based system:

$$\mathbf{z}_k = h\left(\mathbf{x}_k\right) + \mathbf{w}_k. \tag{40}$$

The noise term, $\mathbf{w}_k$ is, as in (23), zero mean and white and follows a normal distribution with covariance matrix $R_k \equiv \mathbb{E}[\mathbf{w}_k, \mathbf{w}_i]$.

*4.3.3. Process Update.* In the time update step of the extended Kalman filter, the mean, $\widehat{\mathbf{x}}$, and the error covariance matrix, $P$, are propagated in time [15, 20]:

$$\begin{aligned} \widehat{\mathbf{x}}_{k|k-1} &= F\widehat{\mathbf{x}}_{k-1}, \\ P_{k|k-1} &= FP_{k-1}F^T + Q, \end{aligned} \tag{41}$$

where $Q$ denotes the process noise covariance matrix. The final noise covariance matrix results as block matrices of $Q_{pv} = G_{pv}VG_{pv}^T$ and $Q_t$ (given by (34)):

$$Q = \left(\begin{array}{c|c} Q_{pv} & \mathbf{0}_{6\times 2} \\ \hline \mathbf{0}_{2\times 6} & Q_t \end{array}\right). \tag{42}$$

*4.3.4. Measurement Update.* The extended Kalman filter approximates (12) as follows [15, 20]:

$$\begin{aligned} \widehat{\mathbf{x}}_k &= \widehat{\mathbf{x}}_{k|k-1} + K\left(\mathbf{z}_k - h\left(\widehat{\mathbf{x}}_{k|k-1}\right)\right), \\ P_{k|k-1} &= \left(I + K_k H_k\right) P_{k|k-1}. \end{aligned} \tag{43}$$

Matrix $K$ is the Kalman gain and is given by $K = P_{k|k-1}H_k\left(H_k P_{k|k-1} H_k^T + R_k\right)^{-1}$. Here, $R_{\rho,k}$ is the measurement covariance matrix that contains a variance estimate estimated by RTKLIB for each pseudorange on its diagonal. These variance estimates are based on the user range accuracy index. If a WLAN position is obtained, the covariance matrix is extended by a diagonal block containing a variance for each coordinate of the WLAN position:

$$R = \left(\begin{array}{c|c} R_\rho & \mathbf{0} \\ \hline \mathbf{0} & R_{\text{WLAN}} \end{array}\right). \tag{44}$$

We found a standard deviation of $\sigma_{\text{WLAN}} = 0.5$ to achieve the lowest errors for both trajectories, although we expected that larger values would model the error of the WLAN maximum likelihood position estimate better [21].

## 5. Experimental Results

To analyze the performance of the particle filter, we conducted experiments for two trajectories. We begin this section with a description of these experiments and continue with some notes concerning the implementation of the particle filter. The outcomes of these experiments are reported subsequently in Sections 5.3 and 5.4.

*5.1. Experimental Setup and Data Recording.* The data used to evaluate our algorithm is from a test bed at the Universidad Autónoma de Querétaro. The test bed constitutes two relatively small buildings (about 8 × 40 meters) with roofed passageways through which the buildings are entered. These two buildings and a larger four-storey building in the south west surround a larger open space. In addition to the roofed passageways, several trees block the view to the sky. Figure 1 depicts the test bed and the two trajectories used to evaluate the algorithm. The first trajectory, named Trajectory-1, commences indoors without GPS observations and


Figure 1: Test bed and trajectories. Circles indicate starting point of the trajectories. (Notice, the trajectory actually followed the existing paths, which are illustrated by densely dotted orange lines. The markers of the trajectories are the positions at which the WLAN data was recorded; they are obtained from the measurements overlaid on the test bed map. The mismatch between the trajectories and the background image is due to errors of the overlay procedure, errors of the experimenter marking the measurement positions on the map, and issues when dealing with geographic data, such as projections and coordinate transformations.)

runs mainly indoors and along the semiopen passageways along the buildings. It enters the right building only once and ends in the open space, nearby some trees. The second trajectory is called Trajectory-2. It starts in the open space and continues on it, passes below trees, and leads to the passageway of the right building. Following that passageway, it enters the right building once and proceeds to the roofed passageway of the left building. From the middle of the left building, it runs around the small lawn area and returns on the passageway to the passageway of the right building, from where it turns sharp right towards the open space. Trajectory-2 ends further southwest on the open space, in front of a four-storey building. Since Trajectory-1 has larger indoor sections, we refer to it as indoor-like trajectory. The opposite is true for Trajectory-2; we call it outdoor-like trajectory.

This localization scenario presents a harsh environment for GNSS, as scattering, blockage and multipath propagation are very likely. The two buildings used in the indoor sections of the experiments are relatively lightweight. They are made of brick walls and the sectioning indoors is principally done by soft-partitions. As the climate favours open doors and windows, these buildings attenuate signals rather little; thus, the indoor environment is challenging for WLAN fingerprinting as well.

We recorded the RSSIs with a laptop with a consumer-grade wireless network interface card. To create the radio maps, RSSIs and MAC addresses need to be captured at certain, known locations. Therefore we modified JMapViewer, a map application providing access to OpenStreetMap data. By a click on the map, at the location that corresponds to that of the experimenter, the position is obtained and the WLAN data capturing process is initialized: RSSIs from all available access points and the corresponding MAC addresses are captured. A second click on the user interface, after three to five seconds, issues a command to compute the arithmetic mean and the variance of the RSSIs from each access point. The capturing process finishes by storing the WLAN data and the corresponding position (obtained by from the first click on the map) in a database.

This software and procedure were also used to record the WLAN data for the test trajectories. This resulted in a stop-and-go motion, which may not be the most common motion of a pedestrian, but this facilitates recording a ground truth. Again, the WLAN data at each position of a trajectory was recorded for a few seconds. The ground truth of both paths is depicted in Figure 1.

The data sets of the trajectories also contain GPS raw data, from which pseudoranges, satellite positions, and pseudorange correction terms were obtained. This data was recorded continuously in time. The used receiver is an u-blox LEA-6T-0 GPS receiver ([22]). To parse and preprocess the data from the GPS receiver, we used RTKLIB.

Time synchronization is achieved by associating a Julian date time-stamp directly after parsing the WLAN and GPS data, respectively. After adding the time-stamps, the data was postprocessed and subsequently stored in the databases. Time synchronization is at least accurate up to half a second, enough to estimate the location of a (walking) pedestrian every second.

During data recording, the experimenter held the laptop in front of his chest, the GPS antenna was mounted on the experimenters head, whereas the WLAN antenna was connected directly to the laptop. The experimenter's body probably influenced the WLAN RSSIs and his head movements may have affected the GNSS data reception. Between the radio map survey phase and localization phase, there are more than 18 months. A degradation of the radio map is likely, nonetheless not severe, because the university's official WLAN infrastructure is relatively stable.

The radio map (see Supplementary Material in [12]) was transformed to a local coordinate frame with an origin at (20.59°, −100.415°, 1800.0 m) and the developed filter was also implemented to compute positions in this frame.

*5.2. Filter Implementation Details.* Before the observation data is processed, it is synchronized with help of the assigned time-stamps.

The reception of all necessary GPS data, especially ephemerides, takes a few seconds. For that reason and the knowledge that a WLAN based location estimate is likely inexact in that area, we skip the first 35 seconds of Trajectory-2 and start the estimation with GPS observations available.

To initialize the filter we distributed the particles uniformly over the test bed and set the velocity to zero. The initial position estimate is close to the centroid of the initial particle distribution and thus relatively close to the centre of the test bed.

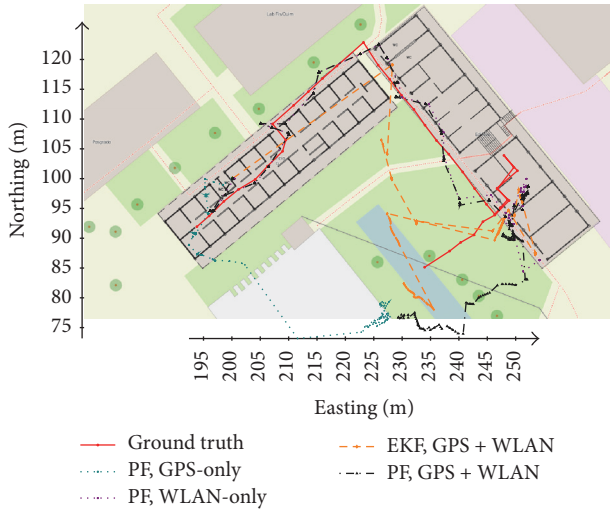



Figure 2: Position estimates of the particle filter in GPS + WLAN, GPS-only, and WLAN-only configuration and of the extended Kalman filter and the ground truth of Trajectory-1.

We repeat the estimation of a trajectory 50 times with random initialization of the particles. The results, that is RMSE, standard deviation, and the data to create the figures, are averages over these 50 repetitions.

The particle filter operates with 1000 particles. This assures a certain reliability of the filter outcomes.

*5.3. Indoor-Like Trajectory: Trajectory-1.* Figure 2 depicts the trajectory estimated by the particle filter (in three configurations), by the extended Kalman filter and the ground truth data. The path estimated by the particle filter follows the ground truth quite well. Visible are jumps of the position estimates. They are caused by a lack of observations. That is because RSSIs are received only every 2 s to 5 s. If, in that time interval, no GPS observations are obtained, the filter can only predict but not correct that prediction, which does not alter the position estimate. In the moment that again a reliable RSSI measurement is received, the filter corrects the prediction and the position estimate jumps to the new corrected estimate. The last section consists of an indoor part immediately followed by an outdoor part. When the trajectory enters the building (around (247, 94) m), the estimated location is already off by a few meters and also the path inside the room is not that accurately estimated. After leaving the room, the estimates drift further and the error increases. During the last section of the trajectory, the particle filter estimates follow the ground truth poorly with an offset. The extended Kalman filter shows a very poor performance. The trajectory shows large position jumps. A comparison of the extended Kalman filter estimates with that from the particle filter suggests that the extended Kalman filter takes the position estimate from WLAN not much into account; if it does, these large jumps occur. This was not expected, because the variance of the WLAN position estimate is set lower than former experiments suggested. Furthermore, the estimates of the extended Kalman filter on the roofed passageway of the

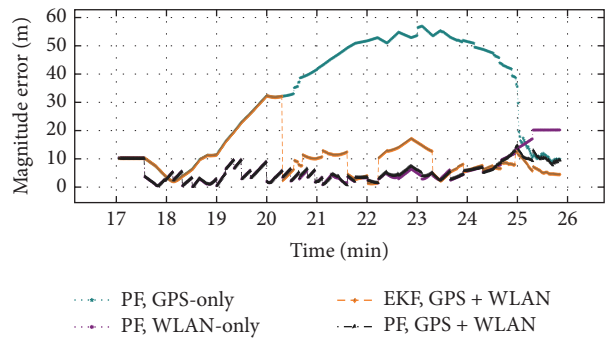



Figure 3: Error of the particle filter in GPS + WLAN, GPS-only, and WLAN-only configuration and of the extended Kalman filter.

right building are pulled into the open space. This indicates that the extended Kalman filter weights the GPS information much more than the particle filter does, even though the GPS data are apparently of low quality. Only towards the end of the path, when outdoors on the open space, the extended Kalman filter estimates get better, even better than those of the particle filter.

The availability of sensory data confirms the described behaviour. Information about the availability can be deduced from Figure 3. This figure depicts the magnitude error over time for the three particle filter configurations and for the extended Kalman filter. The error of the WLAN stand-alone solution indicates that WLAN data is available almost all the time. The error is low, except after the initialization, when no measurements are received for some seconds, and during the outdoor section, where WLAN location fingerprinting is known to perform poorly. The GPS stand-alone solution yields low errors during the first minutes or so, although no GPS data is received. This is due to the initial position that is relatively close to the true start position. The similarity of errors of the particle filter hybrid solution and the WLAN-only solution implies that no GPS data was received for the largest time of the trajectory, or it at least implies that GPS information was not fused into the hybrid solution.

Taking into consideration that WLAN data was available all the time and GPS-only towards the end, the accuracy of the hybrid solution can be explained in more detail: up to minute 24.5, the estimates are governed by the RSSI observations. Therefore, the errors are small where fingerprints can be distinguished well, indoors and most of the time along the passageways. (The reader is referred to [21] for a performance evaluation of the (Gaussian process regression based) WLAN-only location estimator.) The large errors during the last minutes can be explained as follows. After a few seconds outside of the right building, GPS data are received and the filter integrates GPS observations into the solution. The error of the GPS stand-alone solution is still high but decreases as the solution converges to the true location. Simultaneously, while the mobile terminal departs further from the building, the radio map data becomes more ambiguous, which increases the WLAN-only error. The error of the hybrid solution increases because GPS data are still inaccurate, they are still converging to the true location, and

Table 1: Average RMS error and standard deviation of the particle filter in GPS + WLAN, GPS-only, and WLAN-only configuration and of the extended Kalman filter for Trajectory-1.

| | PF, GPS + WLAN | PF, GPS-only | PF, WLAN-only | EKF, GPS + WLAN |
|---|---|---|---|---|
| RMSE (m) | 6.21 | 37.33 | 7.80 | 12.97 |
| Avg. SD (m) | 5.81 | 48.27 | 6.02 | 2.35 |

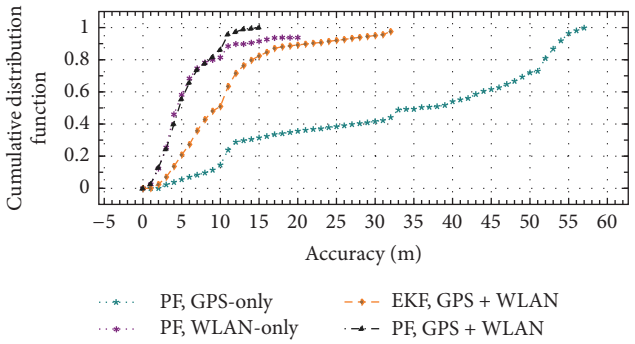



Figure 4: Empirical error distribution function of the particle filter in GPS + WLAN, GPS-only, and WLAN-only configuration and of the extended Kalman filter for Trajectory-1.

WLAN data are becoming inexact. The particle filter weights the WLAN data more than the GPS data until minute 25. Then the GPS observations get more accurate and the particle filter begins weighting the GPS observations more than the WLAN observations. This is the desired effect; as one can see, during the last minute, the GPS-only solution yields higher accuracy than the WLAN stand-alone solution. Shifting the weights to GPS observation induces a decrease of the GPS + WLAN solution's error. Notice that the hybrid solution's error is always equal or lower than the error of the better performing solution, pointing to an adequate weighting of the two information sources. However, the GPS data, on which the hybrid solution is relying during the last minute, is still relatively poor, maybe because the filter has not yet converged after the huge deviations, or because the last part of Trajectory-1 is surrounded by trees.

The error of the extended Kalman filter is initially equal to that of the GPS-only configuration. It takes about three minutes until a WLAN position estimate is considered by the extended Kalman filter. In that moment, the error decreases to the magnitude of the error of the particle filter. While the particle filter solution mainly ignores the GPS information, the extended Kalman filter occasionally weights the GPS data and deviates from the quite accurate WLAN-only solution. A better performance can be seen at the end of the path, when GPS data quality increases due to better visibility of the satellites. The GPS measurements get more weight while the WLAN data is weighted less. During the last minute, the extended Kalman filter outperforms the particle filter.

The precision of the algorithm can be assessed from Figure 4, which illustrates the empirical cumulative distribution function of the errors. The hybrid solution presents the highest accuracy and precision, with a median accuracy of 5 m and an accuracy of 11 m at 90% probability.

The extended Kalman filter solution yielded an accuracy of 10 m at 50% and 20 m at 90% probability. That is about twice as low compared with the accuracy of the particle filter. The maximum error of the extended Kalman filter solution is about 30 m, whereas the maximum error of the particle filter is 15 m.

Table 1 compares the performance of the particle filter hybrid solution with the individual GPS and WLAN solutions and with the extended Kalman filter numerically, that is, in terms of the root-mean-square error (RMSE) and the standard deviation (SD). The hybrid solution clearly outperforms the two other particle filter configurations. The GPS-only solution is very inaccurate, because GPS observations were only available during the last section of the trajectory. For the same reason, the WLAN stand-alone solution is almost as good as the hybrid solution. Nevertheless, it has a higher probability for large errors. Due to the additional information in terms of GPS observations in the last section of the trajectory, the hybrid solution is more accurate than the WLAN-only solution. The RMSE of the particle filter WLAN + GNSS configuration is in comparison with the extended Kalman filter solution two times smaller. However, considering the standard deviation of these two filters, the outcome of extended Kalman filter is two times smaller than that of the particle filter.

The error distribution function and the standard deviation of the particle cloud show that the hybrid solution is not only more accurate but also more precise; its empirical error distribution function presents the lightest tails. The GPS + WLAN configuration also yielded the smallest standard deviation.

*5.4. Outdoor-Like Trajectory: Trajectory-2.* The overall performance of the filter for Trajectory-2 is depicted in Figure 5. The continuous availability of GPS observations is observable; the stop-and-go motion causes the position estimates to cluster, because, during the phases without motion, the filter continues estimating position of the static mobile terminal based on GPS observations. In between these clusters, the estimated position tends to jump, which can be explained again by the reception of a reliable RSSI observation after some time. After the estimates converge closer to the starting point, the estimated path follows roughly the ground truth. The estimates become more exact where the mobile terminal enters the right building. However, most times along the roofed passageways and around the small lawn area the localization performance is only fair. More exact estimates can be seen about the corner, where the two buildings almost touch, and later on the open space again.

The extended Kalman filter shows a similar positioning performance. It estimates well the entering of the right

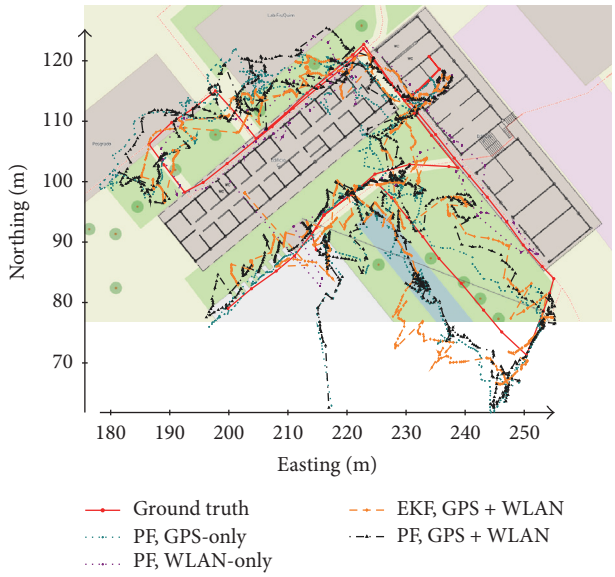



Figure 5: Particle filter, extended Kalman filter estimates, and ground truth of Trajectory-2. Results of the particle filter are shown for the three configurations GPS + WLAN, GPS-only, and WLAN-only.

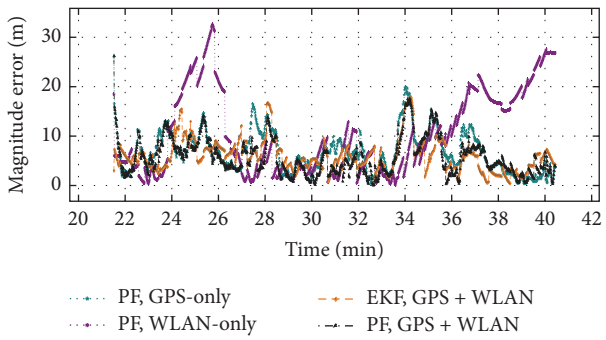



Figure 6: Error of the particle filter in GPS + WLAN, GPS-only, and WLAN-only configuration and of the extended Kalman filter for Trajectory-2.

building. On the semiopen passageway of the left building and around the small lawn area, the accuracy decreases as well; nevertheless, position estimates of the extended Kalman filter appear smoother than those of the particle filter. The accuracy of the extended Kalman filter estimates increase again towards the end of the path on the open space.

To reason about the causes of the described performances, we refer to Figure 6. It illustrates the magnitude error, again for the three configurations of the hybrid algorithm and the extended Kalman filter. This figure shows that the error of the particle filter GPS + WLAN solution is in average below 5 m; nevertheless, in various moments, it exceeds 10 m. The GPS-only solution yields errors similar to the hybrid solution and errors smaller than that of the WLAN stand-alone solution. The outcomes based on only WLAN are very poor, especially about 24 min to 26 min and for the last 4 min of the trajectory.

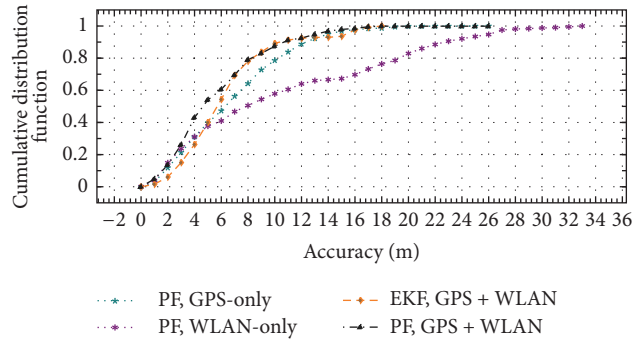



Figure 7: Empirical error distribution function of the particle filter in GPS + WLAN, GPS-only, and WLAN-only configuration and of the extended Kalman filter for Trajectory-2.

The low accuracy of the particle filter's GPS + WLAN solutions about 24-25 min is because of the poor quality of WLAN and GPS observations. During that time interval, the WLAN observations are very inexact for which reason they are basically unregarded, and therefore the accuracy of the hybrid solution equals that of the GPS-only solution. Between minutes 34 and 35, the poor accuracy of the hybrid solution is caused by an inappropriate weighting of the observations. The filter is too confident about the GPS observations or estimates the quality of the WLAN observations as too poor, respectively. The same aspect is visible at the beginning around 23 min and also about 27.5 min. In general, however, the accuracy of the hybrid solution, again, leans towards the more accurate sensor information; it never presents the largest errors among the three configurations. Figure 6 indicates that the weighting mechanism works reasonably well besides a few sporadic exceptions.

The accuracy of the extended Kalman filter is also most of the time about 5 m and shows only a few peaks that are larger than 10 m; see, for example, minutes 24, 28, and 34. It is worth noting that in some occasions the extended Kalman filter presents the lowest errors; refer to minutes 25, 35, 37, and 38. In general, Figure 6 confirms the similarity of the accuracy of the particle filter in GPS + WLAN configuration and the extended Kalman filter.

We examine again the cumulative distribution function of the error, as shown in Figure 7. The proposed algorithm yielded a median accuracy of 4.5 m and only 10% of the errors are larger than 10 m. As the hybrid solution makes use of both sensor data, it is not surprising that it outperforms the individual solutions. The median error for GPS-only is roughly 6.5 m and that of the WLAN-only configuration is 8 m. The difference is even bigger at 90% probability; GPS-only yielded 12 m and WLAN-only 23 m. Thus, in comparison with the two stand-alone solutions, the integration of the two systems improves also the precision.

The median accuracy of the extended Kalman filter is 5.5 m, only one meter larger than that of the particle filter hybrid solution. The error at 90% probability is, as for the particle filter in GPS + WLAN configuration, 10 m. The maximum error of the extended Kalman filter is about 18 m, that is almost 10 m less than the maximum error of the hybrid

Table 2: Average RMS error and standard deviation of the particle filter in GPS + WLAN, GPS-only, and WLAN-only configuration and of the extended Kalman filter for Trajectory-2.

| | PF, GPS + WLAN | PF, GPS-only | PF, WLAN-only | EKF, GPS + WLAN |
|---|---|---|---|---|
| RMSE (m) | 6.68 | 7.88 | 13.36 | 6.49 |
| Avg. SD (m) | 2.70 | 2.99 | 4.81 | 1.04 |

solution. However, the largest error of the particle filter stems clearly from the initial estimate, before the first observations were obtained; the maximum error of the extended Kalman filter occurred about 34 min.

Table 2 summarizes the performance analysis and confirms the previously made notions. It presents the RMS error and the standard deviation derived from the particle cloud and contrasts them for the three configurations, as well the RMSE and the standard deviation of the extended Kalman filter are shown. The extended Kalman filter yielded the smallest RMS error and standard deviation, although the GPS + WLAN configuration's median error is smaller than the median error of the extended Kalman filter. The hybrid solution of the particle filter is in average slightly more inaccurate and its standard deviation is more than twice than that of the extended Kalman filter. As expected, the particle filter in GPS + WLAN configuration outperforms the GPS-only and WLAN-only solutions. The GPS stand-alone solution performs also well considering the localization scenario. The performance based on only WLAN observations is mediocre, because large parts of the paths are outdoors, where fingerprints are ambiguous.

The stated error figures are specific for that system and scenario and shall be put into context. If the measurements of GPS and WLAN are severely degraded, the particle filter can only yield poor estimates. An idea about the quality of the measurements can be gained from Figure 8.

Figure 8 compares the estimates for Trajectory-2 of the GPS-only particle filter solution with that of the commercial GPS receiver ([22]) and with the least squares (LS) solution (that was used to estimate the clock offset of the receiver). Additionally, this figure depicts a second trajectory from the commercial GPS receiver from another experimental run. The black trajectory is derived from the same GPS observations that were fed into the particle filter (teal curve) and least square algorithm (cyan curve). The grey trajectory is an alternative trajectory estimated by the GPS receiver and was recorded on the same path, but during another experiment; the experimenter's motion during this experiment was continuous.

The commercial GPS receiver estimates (black path) the trajectory well at the beginning, until the path enters the right building. The opposite holds for the GPS-only solution of the particle filter; its estimates are pulled towards the open space. The indoor part is not recognized by the GPS receiver, whereas the outcomes of the GPS stand-alone solution are good in the indoor section. On the roofed passageway of the left building, only the GPS receiver provides more or less adequate estimates; the estimates appear of higher accuracy than that of the GPS-only solution. Later, on the passageway of the right building, the GPS receiver's accuracy decreases

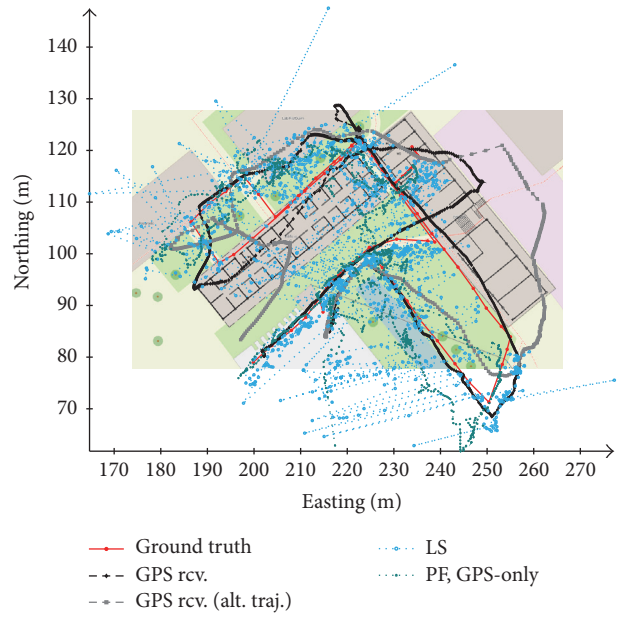



Figure 8: Position estimates of the GPS-only solution, the GPS receiver, and the least squares solution. The black (GPS rcv.) and the teal trajectory (PF, GPS-only) were obtained from the same experimental run. The grey trajectory (GPS rcv.) was recorded during a different experiment along the same path, though, without the stops.

again; its estimates cross the right building until they come back on track which in the receiver is on the open area. The GPS-only solution shows better results on the passageway of the right building, but on the open space the GPS receiver appears to be more accurate again.

The least squares solution illustrates the quality of the GPS measurements and the alternative trajectory of the GPS receiver shows the harshness of the environment. The estimates of the least squares solution are similar compared to that of the GPS-only solution, but much more noisy due to the missing motion model. Nonetheless, position fixes during the indoor part are surprisingly good. The alternative GPS trajectory is worse than the black trajectory, especially on the semiopen passageway of the right building and after surrounding the small lawn area; no position fixes could be estimated during the last minutes of the trajectory.

A comparison of the accuracy of the trajectory of the GPS receiver and the hybrid particle filter is shown in Figure 9. That figure depicts the magnitude error of Trajectory-2 obtained from the hybrid particle filter next to the error from the GPS receiver. Additionally, we show the number

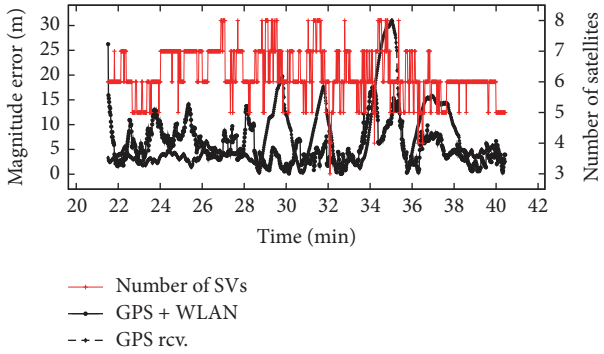



Figure 9: Magnitude error of hybrid GPS + WLAN solution and of GPS receiver.

of satellites. At the beginning of the trajectory, on the open space, the GPS receiver is more accurate than the hybrid solution. It actually follows the ground truth very well, even on the passageway of the right building, which is remarkable under these conditions (it might even be an exception as the alternative trajectory of the GPS receiver suggests). The indoor section and the path next to the left building, in general after minute 29, are nevertheless estimated poorly. Errors rise up to 20 m and 30 m. These large errors occur when the number of available satellites is varying a lot, indicating quick changes of the sky view which degrades GPS performance. Time intervals of good GPS performance correlate with sections where the number of observations is constant.

For most of the time of the trajectory, 6 to 7 GPS observations are available, a fair amount, given the scenario. Only in a few epochs, the number of satellites decreases to four and only one time it is below four. That means, that during time intervals of poor GPS performance, the number of observations was fair enough to get a good position fix, but that the quality of the observations was not sufficient for accurate position estimates.

The integration of WLAN data improves the localization accuracy. The GPS receiver's RMSE for that trajectory is 9.79 m, about three meters larger than the RMSE of the particle filter and of the extended Kalman filter.

## 6. Discussion

The performance of GNSS degrades in areas with obstructed satellite-to-receiver link, because, for example, scattering and multipath effects delay the signal and increase the propagation time. However, the same obstructions attenuate WLAN signals so that the spatial RSSI distribution is rather heterogeneous. This facilitates distinguishing nearby RSSIs and improves the resolution and the accuracy of a WLAN fingerprint localization system. The opposite effect occurs in open areas, where GNSS's position fixes are usually accurate, but position estimates relying on WLAN RSSI often show large errors. These large errors are caused by very similar RSSI patterns at distant locations.

As expected, the particle filter hybrid solution attains the highest accuracy and precision compared with the stand-alone solutions. The smaller error of the two stand-alone configurations presents an upper bound for the error of the hybrid solution.

In comparison with the extended Kalman filter, the proposed algorithms outperformed the extended Kalman filter in the indoor-like scenario. For Trajectory-2, the performance of the particle filter and the extended Kalman filter are comparable, the particle filter performed in average slightly worse than the extended Kalman filter. The extended Kalman filter estimates are more precise than the estimates of the particle filter. This is because of the little amounts of noise that are introduced in the process model of the particle filter to mitigate sample impoverishment (caused by resampling). However, the extended Kalman filter's performance is considerably poorer (large position jumps) than that of the particle filter when only WLAN data was available. The integration of WLAN data works better in the particle filter. From our point of view the disadvantage that the extended Kalman filter presents in scenarios without or with few GPS data does not outweigh the small advantage shown in the scenario where GPS and WLAN data were basically available all the time.

Sections of low accuracy of the hybrid solution are due to the loss of one of the systems or due to very poor quality of one of the observations in combination with sharp direction turns and/or changes of the sky view. As for most of the time one of the systems operates reasonably well, extreme deviations from ground truth, as it is common for fusion on a position level and can be seen for the extended Kalman filter for Trajectory-1, can be avoided, because the continuous adaption of the weights results in smooth location estimates, also during indoor/outdoor transitions.

The chosen motion model is very general and versatile and served well in our experiments. However, an adequate model must be chosen according to the final application.

One drawback of our method is the computational complexity of the RSSI interpolation. Predicting RSSIs requires the inversion of the covariance matrix, which has a dimension equal to the number of used particles. One way to reduce the computational costs is to reduce the number of particles. To compensate the loss of accuracy due to reduction of particles, a better importance density is advisable. For some scenarios, an extended Kalman filter solution can achieve similar or better performance at lower computational complexity. Our methodology used a maximum likelihood estimator to estimate a position from WLAN RSSI; this is based as well on the Gaussian process regression. Replacing the used WLAN position estimation with a simpler algorithm (e.g., $k$-Nearest Neighbour) would reduce the complexity of the extended Kalman filter further. The concrete impact of the above-mentioned changes on the positioning accuracy of the particle filter requires further research.

The ubiquity of the proposed method depends heavily on the dissemination of WLAN fingerprinting location systems, more precisely on the coverage with radio maps, and that they are accessible and compatible. For further thoughts on that we refer to [4]. Another point we are presuming is the accessibility to the pseudoranges of a GNSS receiver. However, most

off-the-shelf receivers do not provide pseudoranges yet. This is, however, changing right now as smartphones with access to raw measurement recently appeared on the market.

One objective of this study was the exploitation of the contrary information provided by pseudoranges on the one hand and RSSI on the other hand. We consider the automatic weighting of the different sensor information as functioning, but it is also apparent that it is improvable. Some relatively simple and straight forward ideas and changes in the presented system are likely to improve the localization accuracy.

The simplest change is the use of a multiconstellation, multifrequency receiver. This improves the coverage because the use of those receivers increases the number of receivable signals, and it improves the localization performance because a second frequency enables a more accurate correction of GNSS errors. Exploiting the carrier phase information would significantly decrease GNSS positioning error; nevertheless, such receivers are just starting to penetrate the market. Additional improvement is expected from multipath mitigation methods in the receiver, for example, at the correlator stage, if these are not yet implemented in the used receiver. Multipath mitigation based on the RSSI data is also worth investigating.

On the WLAN side, the most crucial issue is the choice of the variance in the RSSI likelihood function. It appears advisable to consider time-dependent variance estimates that express the uncertainty of the WLAN observations and not of the predictions of the Gaussian process model. Combining the currently used RSSI variance, containing spatial information, and a RSSI variance estimated from the observed samples, containing temporal information, improves very likely the weighting between the two information sources. Approaches to reducing the computational costs of the interpolation of WLAN RSSI are another open point.

Dependent on the use case, one could create fingerprints only on the paths. This indirectly includes information of the environment and improves the localization performance by removing potentially ambiguous fingerprints. This idea also translates to the interpolation with Gaussian processes, because the variance in areas without fingerprints (training data) will be higher.

## 7. Conclusion

This study presents a new algorithm combining GPS pseudoranges and WLAN RSSIs. The proposed algorithm achieves accurate and robust seamless localization as it exploits the complementary information contained in the sensory data. We developed a likelihood function that overcomes the drawback of the spatially discrete fingerprints by interpolating RSSIs through Gaussian process regression. The integration of the RSSI likelihood function and the pseudorange likelihood function into the particle filter yields a fine grained, automatic weighting of GPS and WLAN observations, which improves the accuracy and precision when compared with the stand-alone solutions using only GPS or WLAN data. This gain in accuracy and precision comes from an increase in computational complexity. The extended Kalman filter used as benchmark, that integrates a WLAN position estimate with GPS pseudoranges, achieved in one scenario similar positioning performance compared to the proposed particle filter. However, without enough GPS data, the extended Kalman filter accuracy degrades much, which is why we consider the particle filter worth its complexity.

In realistic indoor/outdoor experiments, in an environment harsh for GNSS *and* WLAN fingerprinting localization, the proposed method achieved an average positioning accuracy about 7 m. The proposed integration approach of GPS pseudoranges and WLAN RSSI is of low-cost and can improve localization performance of mobile platforms, whenever pseudoranges and RSSIs are available in the navigation processor.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This research was partially supported by the Consejo Nacional de Ciencia y Tecnología and the División de Investigación y Posgrado de la Facultad de Ingeniería de la Universidad Autónoma de Querétaro. The authors thank Karla Alejandra Rojas Camargo for the assistance in programming the data logger and data acquisition. Philipp Richter would also like to thank Eduardo Castaño-Tostado for his comments that improved this work.

## References

[1] E. D. Kaplan and C. J. Hegarty, Eds., *Understanding GPS: Principles and Applications*, Artech House Publishers, 2nd edition, 2006.

[2] N. Samama, "Global Positioning: Technologies and Performance," *Global Positioning: Technologies and Performance*, pp. 1–419, 2007.

[3] N. El-Sheimy and C. Goodall, "Everywhere navigation integrated solutions on consumer mobile devices," *Inside GNSS*, vol. 6, no. 5, pp. 74–82, September 2011.

[4] P. Richter, M. Toledano-Ayala, G. M. Soto-Zarazuá, and E. A. Rivas-Araiza, "A survey of hybridisation methods of GNSS and wireless LAN based positioning system," *Journal of Ambient Intelligence and Smart Environments*, vol. 6, no. 6, pp. 723–738, 2014.

[5] S. Zirari, P. Canalda, and F. Spies, "WiFi GPS based combined positioning algorithm," in *Proceedings of 2010 IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, pp. 684–688, Beijing, China, June 2010.

[6] D. Fernandez, F. Barcelo-Arroyo, I. Martin-Escalona, M. Ciurana, M. Jofre, and E. Gutierrez, "Fusion of WLAN and GNSS observables for positioning in urban areas: The position ambiguity," in *Proceedings of 16th IEEE Symposium on Computers and Communications, ISCC'11*, pp. 748–751, grc, July 2011.

[7] Y. Ma, X. Chen, and Y. Xu, "Wireless local area network assisted GPS in seamless positioning," in *Proceedings of 2012*

*International Conference on Computer Science and Electronics Engineering, ICCSEE 2012*, pp. 612–615, chn, March 2012.

[8] K. Nur, S. Feng, C. Ling, and W. Ochieng, "Integration of GPS with a WiFi high accuracy ranging functionality," *Geo-Spatial Information Science*, vol. 16, no. 3, pp. 155–168, 2013.

[9] B. Li and K. O'Keefe, "WLAN TOA ranging with GNSS hybrid system for indoor navigation," in *Proceedings of In Proceedings of the 26th International Technical Meeting of The Satellite Division of the Institute of Navigation*, pp. 416–425, 2013.

[10] P. Richter, J. Seitz, L. Patiño-Studencka, J. G. Boronat, and J. Thielecke, "Wi-Fi azimuth and position tracking using directional received signal strength measurements," in *Proceedings of 2012 Workshop on Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pp. 72–77, Bonn, Germany, September 2012.

[11] G. Hejc, J. Seitz, and T. Vaupel, "Bayesian sensor fusion of Wi-Fi signal strengths and GNSS code and carrier phases for positioning in urban environments," in *Proceedings of 2014 IEEE/ION Position, Location and Navigation Symposium, PLANS 2014*, pp. 1026–1032, usa, May 2014.

[12] P. Richter and M. Toledano-Ayala, "Revisiting gaussian process regression modeling for localization in wireless sensor networks," *Sensors (Switzerland)*, vol. 15, no. 9, pp. 22587–22615, 2015.

[13] C. E. Rasmussen and C. K. I. Williams, *Gaussian Process for Machine Learning*, The MIT Press, Boston, Mass, USA, 2006.

[14] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*, Cambridge University Press, New York, NY, USA, 2003.

[15] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, 2003.

[16] J. Vermaak and A. Blake, "Nonlinear filtering for speaker tracking in noisy and reverberant environments," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001 (ICASSP'01)*, pp. 3021–3024, USA, May 2001.

[17] T. Takasu, *RTKLIB Manual*, 2013.

[18] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, John Wiley & Sons, Inc., 2001.

[19] M. Weyn, *Opportunistic seamless localization [Ph.D. thesis]*, 2011.

[20] R. G. Brown and P. Y. C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*, vol. 4, John Wiley & Sons, Inc, 2012.

[21] P. Richter, A. Pena-Torres, and M. Toledano-Ayala, "A rigorous evaluation of Gaussian process models for WLAN fingerprinting," in *Proceedings of International Conference on Indoor Positioning and Indoor Navigation, IPIN 2015*, can, October 2015.

[22] u. blox, *u-blox 6 Receiver Description*, 2013.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 4043237, 11 pages
https://doi.org/10.1155/2017/4043237

*Research Article*

# Assessment of Smartphone Positioning Data Quality in the Scope of Citizen Science Contributions

**Angel J. Lopez,[1,2] Ivana Semanjski,[1] Sidharta Gautama,[1] and Daniel Ochoa[2]**

[1] *Department of Telecommunications and Information Processing, Ghent University, St-Pietersnieuwstraat 41, 9000 Ghent, Belgium*
[2] *Facultad de Ingeniería en Electricidad y Computación, Escuela Superior Politécnica del Litoral (ESPOL), Campus Gustavo Galindo, Km 30.5 Vía Perimetral, P.O. Box 09-01-5863, Guayaquil, Ecuador*

Correspondence should be addressed to Angel J. Lopez; angel.lopez@ugent.be

Received 24 February 2017; Revised 8 May 2017; Accepted 24 May 2017; Published 21 June 2017

Academic Editor: Liang Chen



Human travel behaviour has been addressed in many transport studies, where travel survey methods have been widely used to collect self-reported insights of daily mobility patterns. However, since the introduction of Global Navigation Satellite Systems (GNSS) and more recently smartphones with built-in GNSS, researchers have adopted these ubiquitous devices as tools for collecting mobility behaviour data. Although most studies recognize the applicability of this technology, it still has limitations. These are rarely addressed in a quantified manner. Often the quality of the collected data tends to be overestimated and these errors propagate into the aggregated results providing incomplete knowledge of the levels of confidence of the results and conclusions. In this study, we focus on the completeness aspects of data quality using GNSS data from four campaigns in the Flanders region of Belgium. The empirical results are based on mobility behaviour data collected through smartphones and include more than 450 participants over a period of twenty-nine months. Our findings show which transport mode is affected the most and how land use affects the quality of the collected data. In addition, we provide insights into the time to first fix that can be used for a better estimation of travel patterns.

## 1. Introduction

Understanding human travel behaviour lies at the core of planning of transport services and ensuring development of sustainable communities. Traditionally, this data is collected based on self-reported insights into a person's daily mobility patterns. The self-reporting is usually done in the form of travel surveys or interviews. The drawbacks of these methods are well recognized in literature and include, among others, underreporting of short trips [1–3], overestimation of public transport travel times or underestimation of car travel times [4–6], obtaining incomplete and inconsistent information [7, 8], and rounding travel times and distances [9]. Recently, the availability of affordable Global Navigation Satellite Systems (GNSS) devices and mobile communication has presented a new way of acquiring data for mobility studies, including citizen science wherein volunteers participate in some aspects of mobility and environmental matters [10], among others.

Some examples of the GNSS data applicability to better understand individuals' daily mobility behaviour include implementation of GNSS data, together with geographic information system (GIS) technology and an interactive web-based validation application, to derive and validate trip purposes and transport modes [11]. Achieved results showed good match with the national travel survey findings indicating that the suggested approach might be promising alternative to paper diary based methods. Similarly, Zheng et al. [12] use GNSS data of 45 users, collected over six months' period, and apply supervised learning based approach to automatically infer transport mode and gain understanding on how to recognize transport mode exchange locations. Furthermore, Liu et al. [13] and Pan et al. [14] take a deeper look at spatial and temporal patterns of taxis' trajectories to

investigate interurban land use variations. They classify the study area into six traffic areas closely associated with various land use types (e.g., commercial, industrial, residential, institutional, and recreational) and find that human mobility data collected from location aware devices provides opportunity to derive urban land use information in a timely fashion. Using a similar dataset, Guo et al. [15] take a look at taxi tracks to derive trips' origin and destination locations. Hood et al. [16] develop a route choice model based on the GNSS data collected from smartphone users in San Francisco. They extract alternatives using repeated shortest path searches in which both link attributes and generalized cost coefficients were randomized. Their results show that bicycle lanes were preferred to other facility types, especially by infrequent cyclists. The obtained results are intended to be used for bike network infrastructure planning purposes. Duncan et al. [17] examine use of the GNSS data in objectively measuring and studying the relationship of environmental attributes to human behaviour in terms of physical activity and transport-related activity. Mavoa et al. [18] examine children's independent mobility data and implement sequence alignment to match GNSS and travel diary data. They successfully matched around 60% of all trips in between two datasets. Murakami and Wagner [19] explore the potential of validated GNSS tracks to improve self-reported trip data. By comparing the GNSS tracks and travel diaries, they found that, in general, self-reported trip distances were longer than those observed from the GNSS tracks.

Although most studies recognize the potential of GNSS based approaches for understanding human travel behaviours, it does have its limitations [7, 19]. These are rarely addressed in more detail or a quantified manner. Main reason for this is a general agreement that the GNSS data are more detailed than the self-reported insights, have higher spatial and temporal resolution, have high potential to replace traditional data collection approaches, and thus are a better basis for drawing conclusions on individuals' travel behaviour. Furthermore, the ground truth data that can be used to evaluate the quality of the GNSS based insights are rarely collected. There is a lack of awareness about the data quality of the raw sensor data and how their errors propagate into results. This gives an incomplete view on the levels of confidence of the results and conclusions. The aim of this paper is to deepen the understanding on the GNSS data applicability for mobility studies by providing systematic and quantified insights into collected data representativeness in describing individuals' travel behaviour. To do so, we will report on different GNSS based datasets that include trips made with various transport modes. For these datasets, we provide extensive description of the data collection process, as well as trips' related context, and report on how well they capture the actual travel behaviour. To describe the actual travel behaviour, we rely on data collected by independent sensors and/or GIS based validation procedure. By doing so, we hope to fill in some of the gaps recognized in the existing literature, raise the level of awareness about relevance of data quality reporting for GNSS based travel behaviour studies, and provide valuable reference for future research in this field.

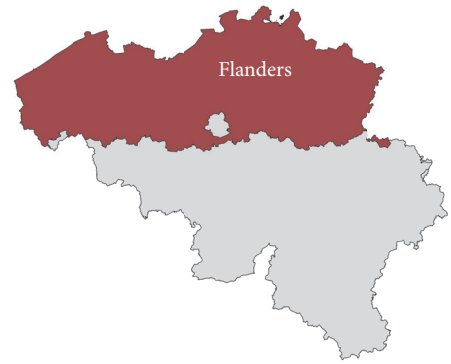



Figure 1: Flanders region in Belgium.

## 2. Method and Data

This paper presents an empirical analysis of the crowd sourcing data collected from four different campaigns that were launched in the Flanders, Belgium (Figure 1). The campaigns were part of different regional and European projects, each one with specific purposes, that are described further in Section 2.3. A common denominator across all projects is one of the data collection methods. This data collection method involved a smartphone application used to collect, among other information, GNSS traces from the participants.

The target population in this study is a generic population and the focus was on sustainable mobility and the transport mode that people employ for commuting and other daily trips. Furthermore, the aggregation level is at trip segment, where trip segment is a trajectory in which a single transport mode is used [20]. Hence, one multimodal trip may contain several trip segments, each travelled by different transport mode. Features of the study area and target population are shown in Table 1.

A description of the reported modes from the participants is depicted in Table 2. In this transport mode categorisation, passenger and driver utilise the same transport mode (e.g., car) but with different roles (i.e., one is driving the vehicle and one is not).

*2.1. Data Representativeness.* In order to have a representative set of mobility behaviour data for whole Flanders population (Figure 1), which is not biased for a single transport mode (i.e., only trips performed with a one specific transport mode), data from four campaigns are combined all together to create a more diverse dataset.

The resulting dataset follows a trip modal split (Figure 2 and Table 3) that is similar to the travel behaviour study conducted by the Department of Mobility and Public Works (Flemish Government). The Flemish research started in 1994 and it is known in Dutch as Onderzoek Verplaatsingsgedrag (OVG), which stands for "Travel Behaviour Survey." The study examines the mobility characteristics of families and individuals and focuses on the mobility behaviour of the Flemish [21]. OVG 4.5 is the last edition of that study and it covers the period from September 2012 to September

Table 1: Study area and target population.

| Study area | Flanders |
|---|---|
| Area size (square kilometres) | 13,522, source: National Committee of Geography of Belgium |
| Population in study area[1] | 6,444,127 |
| Target population | General |
| Transport modes | Foot, bike, drive, passenger, bus, tram, train, and moto |
| Trip activity | Commuting, business related, and recreational trips |
| Data collection period | Feb 2013–Jul 2015 |
| Number of GNSS locations | 10,048,552 |
| Number of travelled kilometres | 71,359 |
| Trip segments | 8,851 |
| Devices (participants)[2] | 457 |

[1]Eurostat. [2]A device or smartphone does not necessarily map a single user, since some devices were shared among the participants.

Table 2: Transport modes collected from the citizen science.

| Transport mode | Description |
|---|---|
| Foot | Participant goes on foot |
| Bike | Bicycle/e-bike as a transport mean |
| Driver | Participant as a car driver |
| Passenger | Participant as a car passenger |
| Bus | Bus as a transport mean |
| Tram | Tram as a transport mean |
| Train | Rail as a transport mean |
| Moto | Two-wheeler vehicle such as scooter and moto |

Table 3: Trip modal split comparison.

| Transport mode | OVG 4.5 | Dataset |
|---|---|---|
| Driver | 51.9% | 46.7% |
| Passenger | 16.9% | 3.1% |
| Bike | 12.8% | 30.1% |
| Train | 1.7% | 2.5% |
| Public transport[1] | 3.5% | 3.6% |
| Foot | 10.8% | 13.3% |
| Others | 2.4% | 0.7% |

[1]Public transport merges data from modes: bus, tram, and metro.

2013 [22]. Therefore, we contrast our dataset with the OVG 4.5 and, apart from passenger and bike modes, all other modes are aligned with the official results. Differences in these two modes can be explained by the way that OVG figures are calculated. The figures are based on the main transport mode; thus subsequential modes are not considered (e.g., going to work may involve modes such as bike, train, and foot, although only train may be reported as a main mode). Nevertheless, our dataset captures all the modes present in the official results of the Flemish region, and, what is more, it reports similar average travelled distance with another study in the region [23], in which passenger and bike modes reported 17 and 4 km, respectively, and in our dataset those modes are 21 and 5 km. Considering this, combined dataset from all four projects is seen as good representation of overall population mobility behaviour.

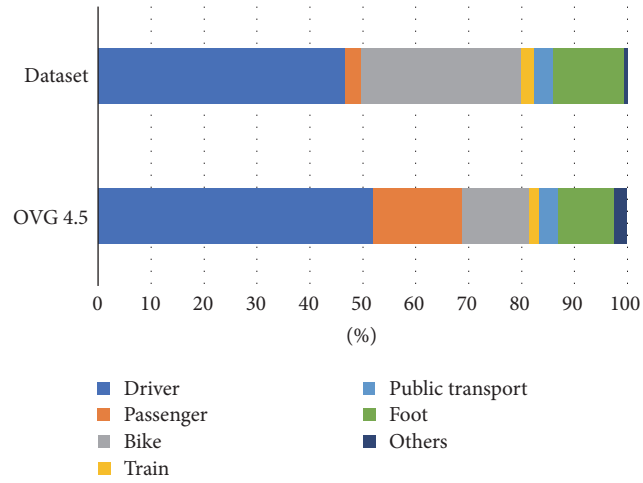



Figure 2: Modal split per trip segment of the crowd sourcing data (dataset) and the Flemish Travel Behaviour Survey (source: OVG 4.5 http://www.mobielvlaanderen.be/ovg/) (OVG 4.5).

*2.2. Mobile Applications.* Data on the campaigns is collected via two Android smartphone applications *Connect* [24] and *Routecoach* [25], which are developed at Ghent University in Belgium. The applications are part of MOVE, a smart city platform for supporting more sustainable mobility behaviour [26]. *Connect* is a mobile application to collect mobility behaviour data. It has two operating ways for gathering data: active and passive mode. Active mode requires the user's intervention to annotate the trip segment information such as purpose, transport mode, and starting/ending of the trip (Figure 3(a)), whereas, in passive mode, data is collected in background without requiring any action of the user. While logging data, trip segments are automatically detected based on stayed points (zones with no movement) and transport modes are classified using sensor data (GNSS and accelerometer sensor); however, the trip purpose is not inferred, but users can annotate their trips later on. In either active mode or passive mode, users can review their travel diary (Figure 3(b)) and correct whether it is needed (e.g., add a trip purpose). By default, *Connect* operates in passive mode but switches to active mode when a trip is started


FIGURE 3: Mobile applications to collect mobility behaviour data: (a) *Connect* trip annotation; (b) *Connect* travel diary; (c) *Routecoach* route suggestion; (d) *Routecoach* leader board.

manually, which means launching the application, filling in the data, and pressing the play button. Thus, there are no visual changes in the graphical user interface (GUI) in either active or passive mode.

Another feature of this application is that it sends survey questionnaires to test the user; such questionnaires are triggered based on some events like the use of certain transport mode [27]. For instance, questions regarding driving behaviour (frequency, accidents, and traffic) can be triggered after validating a certain amount of car mode trips, but such questions are not asked to nonfrequent car users who may under/overestimate some situations; another example of using the survey questionnaires is to collect user's demographic information, and, in this case, a survey is launched once at the beginning of the campaign. The questionnaires are configurable and their content is in function of mobility campaign (i.e., type of behaviour to be captured). Once the survey is filled in by the user, it is no longer triggered. Besides, *Connect* is developed under strict privacy guidelines; consequently, no registration is required to use the application.

The second application, *Routecoach*, shares similar features with *Connect*, but it adds route coaching and gaming features. A main goal of this application is to provide routing information between a defined pair origin/destination points (Figure 3(c)). To do that, *Routecoach* fetches routing information from multiples sources (e.g., the national rail company and the public transport company) to get schedules and routes, which are combined with other transport modes (walk, bike, and car) to suggest a set of feasible routes. Moreover, gaming features were used in the campaign

Table 4: A summary of the datasets per campaign.

| | (i) | (ii) | (iii) | (iv) |
|---|---|---|---|---|
| Number of GNSS locations | 3533752 | 3549218 | 1365198 | 1600384 |
| Number of multimodal trips | 2251 | 2463 | 1315 | 1803 |
| Number of trip segments | 2394 | 2738 | 1814 | 1905 |
| Number of devices | 40 | 18 | 19 | 380 |
| Travelled kilometres | 24737 | 22695 | 11063 | 12864 |
| Data collection period | Jan–Sep 2014 | Feb 2013–Jul 2014 | Mar–Jun 2014 | Jul 2014–Oct 2015 |
| Application name | Connect | Connect | Connect | Routecoach |
| Type of logging | active | active | active | active |
| Sampling frequency | 1 Hz | 1 Hz | 1 Hz | 1 Hz |
| Sensor | GNSS | GNSS | GNSS | GNSS/FUSED[1] |

[1] FUSED locations are not included in this study; a FUSED location is an estimation of the location based on the combination of various sensors such as WIFI, cell networks, GNSS, and Bluetooth.

to reward the most active users and encourage others to participate actively, for example, leader boards for the most biked kilometres (Figure 3(d)) and sustainability challenge board where friends could challenge each other to walk more kilometres during a week. More detailed description of the app and the campaign can be found in [27, 28].

It is worth mentioning that an active mode collects more precise and annotated data than passive mode, yet users might omit short trips (ATM, post office, and bakery) [29]. In contrast, more data is collected in passive mode but such data needs a processing chain (filtering, segmentation, points of interest, map-matching, activity, and mode detection) to be useful [30]. The aforementioned mobile applications work with a sampling frequency of 1 Hz in active mode and variable frequencies in passive mode to increase the time span of the device's battery; such frequencies depend on the battery level (a high battery level has a higher sampling frequency than a low level one).

*2.3. Campaigns.* Three projects made use of the mobile application *Connect* in their campaigns. For data collecting purpose, *Connect* was installed in a set of smartphones that were shared among participants in shifted periods (2-3 weeks). As part of the campaigns, the participants were asked to report their activity using the application in active mode. The following is a brief description of the projects:

(i) Multimodal electric mobility for commuter and business trips (Elmo) investigates whether electric two-wheelers, potentially combined with other types of durable mobility (classic public transport, taxi), could be a valuable alternative for work-related trips, such as commuting and business trip [31].

(ii) Electric vehicles in action (EVA) are a large-scale living lab platform with various types of electric vehicles, charging infrastructure, and data loggers. The purpose of this platform is to check which geographical placement of public charging stations is most fit [27]. The project also aims to assess the impact of electric vehicles on user behaviour, to further support the definition of standards, recommendations, scenarios, and roadmaps for the sustainable deployment of electric vehicles.

(iii) Olympus focuses on networked mobility, aiming at integration between shared mobility (car sharing, carpool, and bike sharing) and private and public transports. It works at different levels of integration: end-users, mobility providers, and supporting services (e.g., charging infrastructure for electrical vehicles). Its campaign started in four Belgian cities (Antwerp, Ghent, Hasselt, and Leuven). In these cities and their stations, electrical shared cars and shared bikes are made available; therefore users can also opt for an electric variant [32].

On the other hand, the *Routecoach* application was a part of sustainable mobility campaign in province of Flemish Brabant. The application was freely available to download and most of the data was collected in passive mode.

(iv) The main aim of the campaign was to develop an evaluation and planning toolkit for mobility projects which is transferable and can be adopted by planners [33]. The data collection process lasted from January to April 2015. In total, 8303 users actively participated by downloading the freely available application and collecting the data on more than 30,000 trips, although, in this study, we only considered the users that manually reported their activity (380 users).

A summary of the collected data is presented in Table 4; some common features across the datasets allow us to merge them into one, for instance, the sampling rate, which is set to 1 Hz when application works in active mode (i.e., users manually report the starting/stop trip). Besides, this mode records a timestamp right after the user's action even if a GNSS location is not fixed.

The dataset (iv) incorporates data from the test period of the application but also postcampaign phase; thus it includes a longer period than the official campaign, though all data collected in passive mode (background data collection) are filtered out since the sampling frequency is not fixed; hence a theoretical estimation of the number of locations could yield an incorrect outcome.

Table 5: Aggregate features from the trip segments.

| Name | Feature | Description |
|---|---|---|
| Collected GNSS | Number of GNSS locations | It is the actual number of collected measurements. |
| Expected GNSS | Theoretical number of GNSS locations | Having a sampling rate of 1 Hz, in theory, the number of GNSS locations to collect is equal to trip segment duration in seconds. |
| TTFF | Time to first fix | Time difference between the trip segment starting time and the first measurement timestamp. |
| Missing GNSS | Number of missing GNSS locations | Difference between the expected and collected GNSS. |


Figure 4: Missing locations in a trip segment.

*2.4. Data Quality.* This study focuses on the completeness aspect of data quality. Consequently, the measurements (GNSS locations) are aggregated into a trip segments' level. Features such as number of collected locations, expected locations, time to first fix (TTFF), and missing locations are extracted from the trip segments (Table 5).

*2.4.1. Missing Locations.* Consider a missing location as a failed event of the GNSS device at getting a fix, so that such events occur during a trip segment. Therefore, a trip segment may include gaps (missing locations) where the location point is not determinate. We represent a trip segment as a list of consecutive locations points with annotations. When a trip segment is reported in active mode, it includes the user's annotations, such as purpose, transport mode, and start/end time. These temporal annotations, start/end time, are independent from the collected locations, since they are recorded right after pressing the play/stop button in the application (Figures 3(a) and 3(c)). Consequently, locations points might be present or not within a trip segment, particularly at the beginning, where a first location can be got after a while (Figure 4). In contrast, when a trip segment is collected in passive mode, the start time matches to the first location timestamp, as well as, the end time and the last location timestamp, because, in passive mode, those labels are inferred by the application using the timestamp of the locations. And yet, locations points may be not present between the first and last location points due to well-known issues of GNSS technology.

Issues like cold/warm start and signal reception can turn out in missing data within a trip segment; hence the resulting segment may include gaps along the trip. To make a distinction among gaps, the missing locations at the beginning of the trip, the gap before the first location, are associated with the cold/warm start issue of the GNSS device, which is the time that the GNSS device needs to acquire the first position after a period of inactivity. This issue often turns out in missing data at the beginning of the trip, especially when a device remains off for long periods [34, 35]. In contrast, the gaps after the first location point are linked to the signal reception issues such as signal loss [36] (e.g., underground travel, bridges, and tunnels) and multipath errors a.k.a. urban canyoning errors [37, 38].

Let $S$ be trip segment represented by tuple $S_j = (P_j, A_j)$, where $P_j$ is a list of location points $P_j = \{p_1, \ldots, p_m\}$ such that $p$ is a location point with a timestamp, $A_j$ is a list of users' annotations $A_j = \{start, end, mode, purpose\}$ with $start < end$, *start* is the starting time of the segment, *end* is the stopping time of the segment, *mode* is the transportation, *purpose* is the trip purpose, and $j$ is a unique identifier of the segment. We assess the quality of the GNSS data using the following equations:

$$\text{Missing locations}_j = \frac{n\left(P_j\right)}{\left|A_{j,\text{end}} - A_{j,\text{start}}\right|} \tag{1}$$

$$\text{TTFF}_j = \frac{\left\lfloor t\left(P_{j,1}\right) - A_{j,\text{start}}\right\rfloor}{\left|A_{j,\text{end}} - A_{j,\text{start}}\right|} \tag{2}$$

$$\text{Missing within trip}_j = \text{Missing locations}_j - \text{TTFF}_j \tag{3}$$

$$\text{Collected locations}_j = 1 - \text{Missing locations}, \tag{4}$$

where $n(\,)$ and $t(\,)$ are functions to count the number of locations in $P_j$ and to extract the timestamp from a location point. Having a sampling rate of 1 Hz, in active mode, the theoretical number of GNSS locations is equal to segment duration in seconds (i.e., difference between *end* and *start* times); thus the proportion of missing locations is equal to the number of collected locations over the theoretical ones (see (1)). The proportion of missing data linked to the time to first fix is the time difference between the first location timestamp

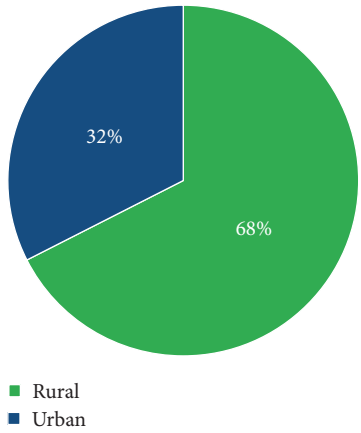



Figure 5: Land use of the trip segment based on the origin area.

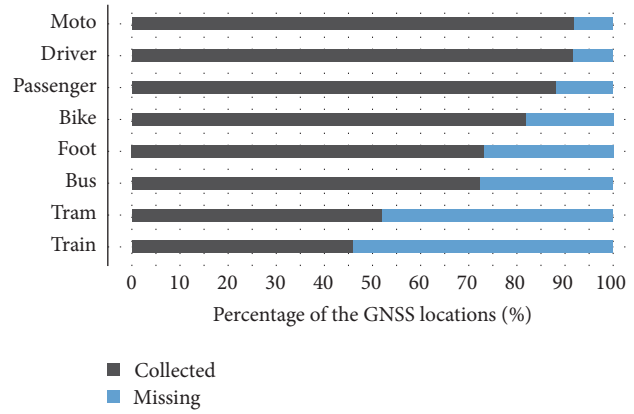



Figure 6: Percentage of missing GNSS locations per trip segment.

and the segment starting time over the theoretical number of locations (see (2)). The proportion of missing locations within a trip segment is the difference between the missing locations and the time to first fix (see (3)). The proportion of collected locations is equal to the missing locations subtracted from the unit (see (4))

*2.4.2. Land Use.* Land use plays an important role in this study; it provides an extra perspective to analyse the GNSS data quality and its relationship to the area where the data is collected; for instance, large structures that are extensively present across urban areas might affect the signal reception differently compared to that in rural areas, where such structures are hardly present. Consequently, a segment is classified as either rural or urban depending on its origin.

The segment origin contains relevant information to assess the missing location due to a cold/warm start effect. Since the time to first fix occurs at the beginning of the segment (Figure 4), the administrative area in which the segment started is used to classify it into rural or urban. Therefore, the administrative areas in Belgium are extracted from *OpenStreetMap contributors* (OSM). OSM is an open access platform for geospatial vector data and it is often considered complete and appropriate for planning studies in comparison to other commercial counterparts [39].

Using a geographic information system, we identify whether a segment lies in a rural or urban area [20], where a spatial operation (interception) is used between the administrative areas and the segments, which turns out in labelled segments based on the land use. A land use share is shown in Figure 5, where more trip segments start from rural areas.

## 3. Results and Discussion

In this paper, we present only results on the GNSS traces collected through smartphone and do not focus our analysis on any other type of data collected in the campaigns.

By comparing the collected GNSS locations and the theoretical number of locations, we calculated the missing data per trip and transport mode (Figure 6). In addition, we observed that *train* mode contains the highest percentage of missing data, collecting less than 50% of the data. To our understanding, a coach makes unfavourable conditions for signal reception because of its dense chassis, like passengers travelling in the low level of a double decker train and underground trajectories [40]. Besides, underground platforms and covered rail stations might affect getting the first fix.

For public transport, *tram* mode behaves slightly better than a train mode, gathering just above 50% of the data. Both transport modes share common things that can cause a signal loss, such as dense chassis, high voltage above them, and covered stations [41]. Instead, *bus* mode collects 72.4% of the data (Figure 6), a notorious improvement over tram, but perhaps not good enough for some applications.

Pedestrians are next on experiencing issues to gather the data; 26.8% of the data are missed on foot mode. This can be explained by user's behaviour for reaching certain destinations. For instance, walking under buildings to avoid rain and sun, taking shortcuts in between narrow corridors, and small stops in stores or covered areas.

Driver and passenger modes, both in car, perform good and collect around 90% of the data. Yet, we expected to hardly see dissimilarity among them, providing it is the same type of vehicle. But it turns out that a car passenger reports slightly less data than a car driver, 88.3% and 91.8, respectively. The differences can be in the interaction with the device, since a car driver must be focused on the road rather than manipulating the smartphone, whereas a passenger is free to interact with the smartphone all way long. These outcomes are comparable to a previous study [42], in which data quality was assessed using GNSS loggers instead. The mentioned study reported 9% of missing data in a car mode. That figure is similar to our findings, where 8.2% of the data are not captured by the smartphone.

Another interesting observation is the two-wheeler vehicles, bike and moto, that report 82% and 92%, respectively. It seems like the road restrictions push moped riders to share the same infrastructure as cars, having a clear view to the satellites most of the time and therefore gathering similar percentage of data. In contrast, bike mode makes use of other

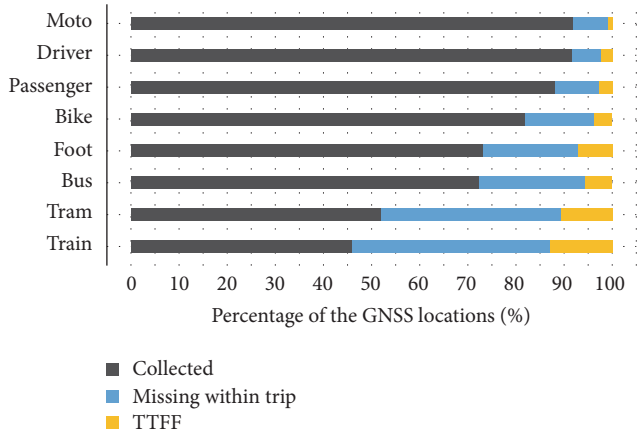



Figure 7: Percentage of missing GNSS locations at the beginning of the trip (TTFF).

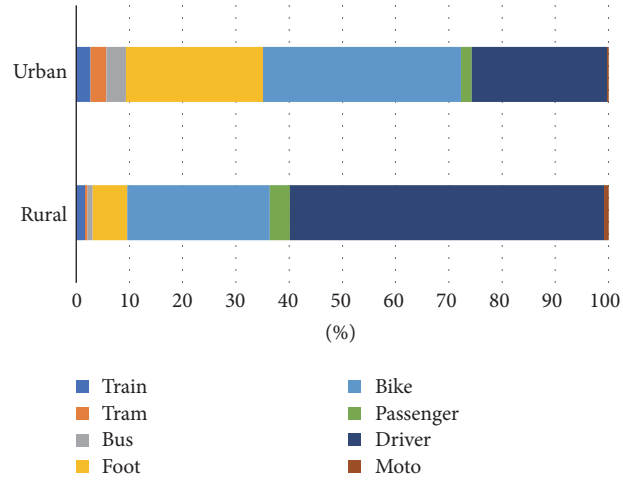



Figure 8: Modal split of the trip segments based on the land use.

road facilities (bicycle lanes and bicycle highway) and it has few restrictions (a biker could ride on a walking path or take same shortcuts as pedestrians). Hence, it faces similar issues for collecting data like pedestrians.

*3.1. Time to First Fix.* Considering the time to first fix as a part of the missing data (Figure 7), we calculated it using the trip start time and the timestamp of the first GNSS location. It turns out that it follows the same trend for all the modes except for on foot mode, in which we observed that not only does the overall data collection struggle in this mode (26.8% of missing data) but also the first fix represents 7.2% of the trip segment.

Both train and tram missed more than 10% of the data before getting the first fix. Having significant gaps at the beginning of the trip leads to travel reporting issues, in which the truth origin of a trip may be far from the first location point, thus underreporting the travelled distance, while the gaps within a segment can be corrected using GIS tools and map-matching techniques. Those techniques align location points with the road network and also interpolate missing locations, yielding a good estimation of the travelled distance.

Additionally, we noticed that travelling in a car as a passenger generates more missing data during the trip than in a car as a driver, since the missing data at the beginning are very close, 2.3% and 2.7% for car and passenger mode, respectively; we can tell the difference is on the interaction with the device.

*3.2. Land Use Effects.* To analyse further the missing data, we considered the land use as factor that could influence the GNSS signal reception. We relied on the information provided by the OSM to extract the administrative areas; thus trips segments were classified as rural and urban depending on trip origin. Results are shown in Figure 8, in which a modal split of the trip count is provided for both urban and rural areas.

Results in Figure 9 were expected for rural and urban areas; that is, urban areas are affected more than rural areas.

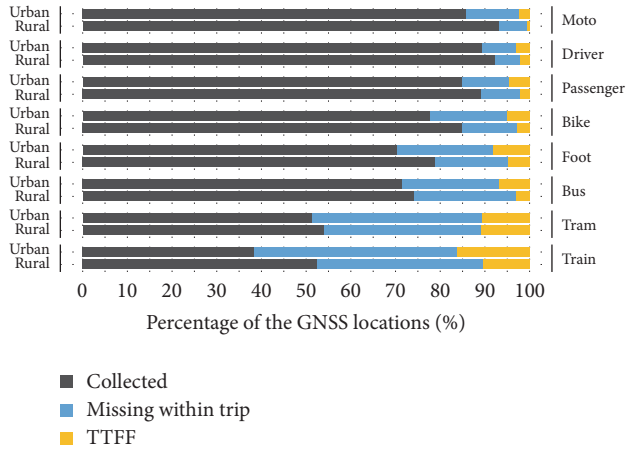



Figure 9: Percentage of missing locations based on the land use.

However, the effect on tram mode seems to be the same in both areas, particularly the time to get the first fix, where we notice similar figures of missing data, 10.8% and 10.7% for rural and urban. Hence, it could be that infrastructure facilities are similar; besides a public transport like tram is not that popular in rural areas, which turns out that those trips are from surrounding urban areas.

Considering the experiment setup, where the sampling frequency is 1 Hz, we can estimate the average delay at getting the first fix. Therefore, we calculate the average travel time per mode and combine it to the percentage of missing data due to the time to first fix (Table 6 and Figure 10), which turns out to be an average estimation of the cold/warm start problem in smartphones.

As was pointed out before, land use exhibits no effect on a tram mode to first fix; in both areas, it reports similar figures. In contrast, modes such as moto, passenger, bike, and bus are increased twofold when these are compared to rural areas (Figure 10).

As an overall outcome, the smartphones reported 84% of the GNSS data; 16% of the data was missed (Figure 11). The

Table 6: An estimation of the time to first fix based on an average travelled time (minutes).

| Transport mode | Travelled time | | Average TTFF | |
|---|---|---|---|---|
| | Average | Std. Dev. | Rural | Urban |
| Bike | 21.9 | 11.0 | 0.6 | 1.1 |
| Bus | 27.0 | 16.1 | 0.8 | 1.9 |
| Driver | 18.6 | 11.1 | 0.4 | 0.5 |
| Foot | 13.7 | 10.5 | 0.7 | 1.1 |
| Moto | 32.9 | 9.6 | 0.2 | 0.8 |
| Passenger | 27.5 | 14.9 | 0.6 | 1.3 |
| Train | 33.2 | 12.7 | 3.4 | 5.4 |
| Tram | 18.6 | 9.7 | 2.0 | 2.0 |

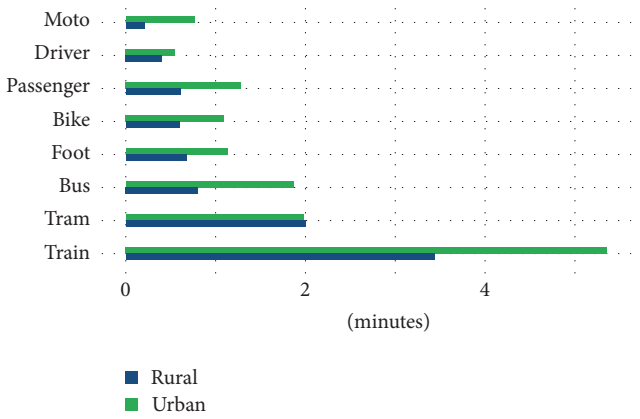



Figure 10: Average time to first fix in smartphones (minutes).

missing data is split in 4% as a part of the cold/warm start problem and 12% is missed within the trip segment (between the first location point and the end of the trip) that can be linked to signal reception issues.

## 4. Conclusions

This paper presents outcomes from four campaigns combined all together, in which the quality of the mobility behaviour data was analysed, more precisely, the completeness aspect of the data. These results can be used as a reference for current and futures mobility studies that use smartphones as a collecting tool, where these figures for missing data can help to put measurements on the time travelled and the distance on multiple transport modes into context. In addition, the results can help to set up parameters on the smartphones applications and to assess the influence on the reporting.

Travelling by train has the most impact on the GNSS reception of the smartphones. However, it may be not that difficult to overcome the lack of data, since that transport mean follows a fix pathway and timetables as well. Therefore, GNSS traces can be aligned with the railway and even dummy-locations can be extrapolated to fill in the gaps. It will require extra steps in the processing chain (map-matching and interpolation) to do this processing, which can lead

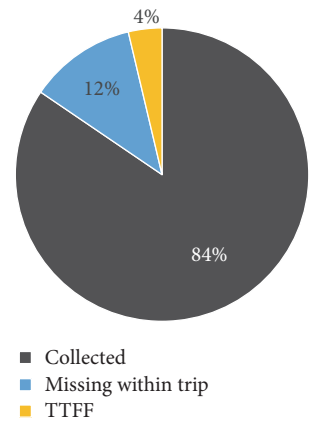



Figure 11: Overall missing data in smartphones as a data collecting tool.

to other types of errors (oversegmentation and misleading trajectories).

It turns out that people travelling in similar vehicles can produce different outcomes based on their role within the vehicle. For example, a person in a car can be either passenger or driver, where drivers reported less missing data than passengers. We have seen that the issue is not related to gathering a first fix during the trip, which leads to the conclusion that the human-device interaction (angle and position) while collecting data has an impact on the quality of the data collection.

Finally, our findings show consistency with a previous study [42], in which a driver mode reported comparable figures (9% of missing data). However, that study made use of GNSS loggers installed in a car instead of smartphones, where the GNSS loggers were mainly affected by the cold/warm start because of the long periods of being turned off (e.g., a car being on a parking lot). In contrast, smartphones are more likely to be working a whole day and, besides, users might be gathering GNSS locations through any other mobile application installed on their devices, which gives a fast response at getting a first fix.

## Disclosure

The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; and in the decision to publish the results.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This research is funded by INTERREG North-West Europe Project New Integrated Smart Transport Options (NISTO),

the Flemish Government Agency for Innovation by Science and Technology, and the Flemish Institute for Mobility.

## References


[1] S. Bricka and C. R. Bhat, "A comparative analysis of GPS-based and travel survey-based data," *Transportation Research Record*, Article ID 1972, pp. 9–20, 2006.

[2] J. Wolf, M. Oliveira, and M. Thompson, "Impact of underreporting on mileage and travel time estimates: results from global positioning system-enhanced household travel survey," *Transportation Research Record*, vol. 1854, no. 3, pp. 189–198, 2003.

[3] S. G. Bricka, S. Sen, R. Paleti, and C. R. Bhat, "An analysis of the factors influencing differences in survey-reported and GPS-recorded trips," *Transportation Research Part C: Emerging Technologies*, vol. 21, no. 1, pp. 67–88, 2012.

[4] C. Carrion and D. Levinson, "Value of travel time reliability: a review of current evidence," *Transportation Research Part A: Policy and Practice*, vol. 46, no. 4, pp. 720–741, 2012.

[5] J. Bates, J. Polak, P. Jones, and A. Cook, "The valuation of reliability for personal travel," *Transportation Research Part E: Logistics and Transportation Review*, vol. 37, no. 2-3, pp. 191–229, 2001.

[6] I. Semanjski and S. Gautama, "Sensing human activity for smart cities' mobility management," in *Smart Cities Technologies*, I. N. Da Silva, Ed., InTech, 2016.

[7] J. Du and L. Aultman-Hall, "Increasing the accuracy of trip rate information from passive multi-day GPS travel datasets: Automatic trip end identification issues," *Transportation Research Part A: Policy and Practice*, vol. 41, no. 3, pp. 220–232, 2007.

[8] P. Stopher, C. FitzGerald, and M. Xu, "Assessing the accuracy of the Sydney Household Travel Survey with GPS," *Transportation*, vol. 34, no. 6, pp. 723–741, 2007.

[9] F. Witlox, "Evaluating the reliability of reported distance data in urban travel behaviour analysis," *Journal of Transport Geography*, vol. 15, no. 3, pp. 172–183, 2007.

[10] Q. Jiang, F. Kresin, A. K. Bregt et al., "Citizen Sensing for Improved Urban Environmental Monitoring," *Journal of Sensors*, vol. 2016, Article ID 5656245, 2016.

[11] W. Bohte and K. Maat, "Deriving and validating trip purposes and travel modes for multi-day GPS-based travel surveys: a large-scale application in the Netherlands," *Transportation Research Part C: Emerging Technologies*, vol. 17, no. 3, pp. 285–297, 2009.

[12] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw GPS data for geographic applications on the web," in *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*, New York, NY, USA, April 2008.

[13] Y. Liu, F. Wang, Y. Xiao, and S. Gao, "Urban land uses and traffic 'source-sink areas': evidence from GPS-enabled taxi data in Shanghai," *Landscape and Urban Planning*, vol. 106, no. 1, pp. 73–87, 2012.

[14] G. Pan, G. Qi, Z. Wu, D. Zhang, and S. Li, "Land-use classification using taxi GPS traces," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 113–123, 2013.

[15] D. Guo, X. Zhu, H. Jin, P. Gao, and C. Andris, "Discovering Spatial Patterns in Origin-Destination Mobility Data," *Transactions in GIS*, vol. 16, no. 3, pp. 411–429, 2012.

[16] J. Hood, E. Sall, and B. Charlton, "A GPS-based bicycle route choice model for San Francisco, California," *Transportation Letters*, vol. 3, no. 1, pp. 63–75, 2011.

[17] M. J. Duncan, H. M. Badland, and W. K. Mummery, "Applying GPS to enhance understanding of transport-related physical activity," *Journal of Science and Medicine in Sport*, vol. 12, no. 5, pp. 549–556, 2009.

[18] S. Mavoa, M. Oliver, K. Witten, and H. M. Badland, "Linking GPS and travel diary data using sequence alignment in a study of children's independent mobility," *International Journal of Health Geographics*, vol. 10, article no. 64, 2011.

[19] E. Murakami and D. P. Wagner, "Can using global positioning system (GPS) improve trip reporting?" *Transportation Research Part C: Emerging Technologies*, vol. 7, no. 2-3, pp. 149–165, 1999.

[20] P. Stopher, E. Clifford, J. Zhang, and C. FitzGerald, *Deducing Mode and Purpose from GPS Data*, Working Paper of the Austrian Key Centre in Transport and Logistics, Institute of Transport and Logistics Studies, University of Sydney, Sydney, Australia, 2008.

[21] OVG, "Flemish Travel Survey," Department of Mobility and Public Works, 2013, Available: http://www.mobielvlaanderen.be/ovg/.

[22] OVG, OVG Flanders 4.5: Travel Survery, 2014.

[23] L. H. Immers and J. E. Stada, Transportation Systems, 2004.

[24] Move., "Connect mobile application," Ghent University. Google Play store, 2013, Available: https://play.google.com/store/apps/details?id=com.move.tripdiary.

[25] Move, "Routecoach mobile application," Ghent University. Google Play store, 2015, Available: https://play.google.com/store/apps/details?id=com.move.routecoach.

[26] A. J. Lopez, I. Semanjski, D. Gillis, J. De Mol, R. Bellens, and S. Gautama, "Development of smart city platform as a tool for supporting more sustainable mobility behaviour," in *Serious Health Games and Apps Conference*, vol. 32, 2015.

[27] S. Vlassenroot, D. Gillis, R. Bellens, and S. Gautama, "The use of smartphone applications in the collection of travel behaviour data," *International Journal of Intelligent Transportation Systems Research*, vol. 13, no. 1, pp. 17–27, 2015.

[28] I. Semanjski, A. J. L. Aguirre, J. De Mol, and S. Gautama, "Policy 2.0 platform for mobile sensing and incentivized targeted shifts in mobility behavior," *Sensors (Switzerland)*, vol. 16, no. 7, article no. 1035, 2016.

[29] D. P. Wagner, "Lexington area travel data collection test: GPS for personal travel surveys," Final Report, Off. Highw. Policy Inf. Off. Technol. Appl. Fed. Highw. Adm. Battelle Transp. Div. Columbus, 1997.

[30] J. Wolf, D. Dr, and R. Guensler, *Using GPS data loggers to replace travel diaries in the collection of travel data*, Georgia Institute of Technology, 2000.

[31] VIM, "Multimodal electric mobility for commuter and business trips," Flanders Institute for Mobility, 2015, Available: http://www.vim.be/projects/elmowork.

[32] Olympus, "Networked mobility solutions, 2015," Available: http://www.olympus-mobility.com.

[33] Nisto, "New Integrated Smart Transport Options," 2015, Available: http://www.nisto-project.eu.

[34] P. Stopher, Q. Jiang, and C. FitzGerald, "Processing gps data from travel surveys," in the 2nd international colloqium on the behavioural foundations of integrated land-use and transportation models: frameworks, models and applications, 2005.

[35] N. Schuessler and K. Axhausen, "Processing raw data from global positioning systems without additional information," *Transportation Research Record*, no. 2105, pp. 28–36, 2009.

[36] G. Draijer, N. Kalfs, and J. Perdok, "Global Positioning System as Data Collection Method for Travel Research," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1719, pp. 147–153, 2000.

[37] N. Schuessler and K. Axhausen, "Identifying trips and activities and their characteristics from GPS raw data without further information," in *Proceedings of the 8th International Conference on Survey Methods in Transport*, Annecy, France, May 2008.

[38] J. Jun, R. Guensler, and J. Ogle, "Smoothing Methods to Minimize Impact of Global Positioning System Random Error on Travel Distance, Speed, and Acceleration Profile Estimates," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1972, pp. 141–150, 2006.

[39] H. H. Hochmair, D. Zielstra, and P. Neis, "Assessing the completeness of bicycle trail and lane features in OpenStreetMap for the United States," *Transactions in GIS*, vol. 19, no. 1, pp. 63–81, 2015.

[40] E. Bertran and J. A. Delgado-Penín, "On the use of GPS receivers in railway environments," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 5, pp. 1452–1460, 2004.

[41] R. Mázl and L. Přeučil, "Sensor data fusion for inertial navigation of trains in GPS-dark areas," in *Proceedings of the 2003 IEEE Intelligent Vehicles Symposium, IV 2003*, pp. 345–350, usa, June 2003.

[42] A. Lopez, I. Semanjski, D. Gillis, D. Ochoa, and S. Gautama, "Travelled distance estimation for GPS-based round trips: car-sharing use case," in *Proceedings of the Fifth International Conference on Data Analytics*, pp. 87–92, DATA ANALYTICS 2016.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 5407605, 11 pages
https://doi.org/10.1155/2017/5407605

*Research Article*

# Feasibility Study of Using Mobile Laser Scanning Point Cloud Data for GNSS Line of Sight Analysis

**Yuwei Chen,[1] Lingli Zhu,[1] Jian Tang,[2] Ling Pei,[3] Antero Kukko,[1] Yiwu Wang,[1] Juha Hyyppä,[1] and Hannu Hyyppä[4]**

[1]*Department of Remote Sensing and Photogrammetry, Finnish Geospatial Research Institute, 02431 Masala, Finland*
[2]*GNSS Research Center, Wuhan University, Wuhan, Hubei, China*
[3]*School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China*
[4]*Department of Real Estate, Planning and Geoinformatics, Aalto University, Espoo, Finland*

Correspondence should be addressed to Jian Tang; tangjian@whu.edu.cn

Received 1 September 2016; Revised 7 December 2016; Accepted 12 February 2017; Published 5 March 2017

Academic Editor: Gonzalo Seco-Granados



The positioning accuracy with good GNSS observation can easily reach centimetre level, supported by advanced GNSS technologies. However, it is still a challenge to offer a robust GNSS based positioning solution in a GNSS degraded area. The concept of GNSS shadow matching has been proposed to enhance the GNSS based position accuracy in city canyons, where the nearby high buildings block parts of the GNSS radio frequency (RF) signals. However, the results rely on the accuracy of the utilized ready-made 3D city model. In this paper, we investigate a solution to generate a GNSS shadow mask with mobile laser scanning (MLS) cloud data. The solution includes removal of noise points, determining the object which only attenuated the RF signal and extraction of the highest obstruction point, and eventually angle calculation for the GNSS shadow mask generation. By analysing the data with the proposed methodology, it is concluded that the MLS point cloud data can be used to extract the GNSS shadow mask after several steps of processing to filter out the hanging objects and the plantings without generating the accurate 3D model, which depicts the boundary of GNSS signal coverage more precisely in city canyon environments compared to traditional 3D models.

## 1. Introduction

GNSS plays an important role in current navigation applications in a pervasive way. However, the accuracy and availability of such solutions in city canyons are a well-known problem, which has attracted researchers' attention in the last few decades. Since currently the urbanization level in developed countries is more than 80 percent and about 50 percent worldwide, the majority of positioning demands are, thus, requested from urban areas. Therefore, augmented solutions helping positioning and navigation in city canyons are obviously needed. The main cause of degraded performance of GNSS in the city canyons is that tall buildings block direct radio signal path from the GNSS to users. Part or even full GNSS observations are missed because of the GNSS shadow cast by the buildings nearby. As a result, the availability and accuracy of positioning are not always guaranteed.

However, GNSS shadow information can be calculated, if 3D city model information is available by applying 3D ray tracing technique [1] to GNSS signals to analyse line of sight (LOS) in the city canyon. The shadow matching concept was first published in 2004 by Tiberius and Verbree [2]. A corresponding GNSS shadow match technique has been evaluated and proved its capability of refining the positioning accuracy by many research groups recently [1–13], especially, researchers from Britain: they proposed the idea and simulated the urban canyon case with multiple GNSS constellations scenario [3] and utilized the 3D city model of London to verify the idea [4, 5] with visibility scoring algorithm [4] to achieve optimized positioning results in London.

Such technology was also to be investigated in Finland [8], United States [9–11], Canada [12], and Taiwan [13]. The major advantage of the shadow matching technology is that it is applicable to receivers which output standard National Marine Electronics Association (NMEA) messages. Thus the potential of the method is extensive, which can be utilized in various platforms to improve the position accuracy in the GNSS signal degrade area, especially for low-end receivers.

There have been several attempts to calculate LOS using various 2.5D surfaces (Digital Surface Model (DSM) or Digital Terrain Model (DTM)). However, all algorithms may rely greatly on the accuracy and integrity of the 2.5D/3D surfaces or models. A 3D model is a simplified version of the real world and some explicit details are omitted deliberately for various reasons. It influences the LOS analysis to some extent. Normally the model accuracy varies between meter-level to decimetre-level [1]. In this research, the shadow mask is generated with centimetre level accurate MLS cloud data instead of any existing 3D models, because we argue that the modelling accuracy degraded during 3D model reconstruction processing from point cloud.

A MLS system consists of a mobile platform, that is, a car, a laser scanner/several laser scanners, and possibly cameras. It is integrated with a georeferencing system consisting of GNSS and inertial measurement unit (IMU). It provides georeferenced 3D point cloud of a measured scene with high accuracy. With good GNSS visibility the errors of MLS point cloud are trivial. Kaartinen et al. [14] demonstrated that the elevation accuracy of commercial and research MLS systems were better than 3.5 cm up to a range of 35 m. The best system achieved a planimetric accuracy of 2.5 cm over a range of 45 m. Applications of MLS include extraction and modelling of buildings [15–19], trees [18, 20–23], pole detection [21, 24–28] and ground and road surface [18, 29–31], and change detection in fluvial environments [32]. Also, Nokia HERE (previously Navteq) True Cars and Google Street View Cars are collecting large data sets; however, no research based on those data sets is released to the public yet.

Processing of MLS data is a relatively new field of science. One of the major disadvantages of the MLS is the limited number of software applications capable of processing the huge amount of data. Current systems can provide a scanning rate of more than 1 Mpts/s. Thus, efficient processing techniques are needed especially when working with raw mobile laser scanning data.

In this paper, a novel solution for GNSS LOS analysis is demonstrated to generate the GNSS shadow mask using MLS point cloud data. The solution includes (1) removal of noise points, (2) removal of points coming from objects that do not interfere with GNSS, such as wires and poles, and determining the objects only attenuating the RF signal, and (3) the highest point extraction and the angle calculation for shadow mask generation. The following section explains why point cloud but not 3D models is utilized for shadow mask generation; Section 3 summarizes the processing algorithms for MLS data, followed by introduction of field tests for the research. Then, we discuss the experimental results and analyse them in detail. Finally, conclusions are drawn and future improvements are discussed.

## 2. Why Point Cloud Rather Than 3D Models

A 3D model is a simplified version of the real world. In order to distinct the difference between dense point cloud and 3D model, the following part is a brief introduction about the procedure of 3D model reconstruction from point cloud. The reconstruction of 3D models of a scene is a complicated process. The universal point cloud processing methods to automatically generate 3D models are still not available. The dedicated modelling methods for different objects are diverse as aforementioned in introduction [15–33]. However, in general, the processing chain includes the following steps as Figure 1 presented: (i) noise point filtering/reduction from the georeferenced point cloud; (ii) object classification: grounds, buildings, roads, trees, and other street furniture such as traffic signs and fences; (iii) building reconstruction by planar detection, edge generalization from zigzagged point cloud to extract the outline of a plane (roof or facade), constrained right-angle processing for all edges; (iv) foreign objects filtering, such as the hanging objects and the plants to minimize the size and complexity of the generated 3D model; (v) meshing or triangulating the building geometry; (vi) texture mapping: project rectified images onto building roofs and facades. With proposed method, shadow mask is available without a specific and detailed 3D city model but raw point cloud, which will save excessive labour cost and system investment.

Compared to original point cloud, it is possible that the accuracy degrades during the model reconstruction in each aforementioned step. Figure 2 shows an example as part of the model reconstruction. As it can be observed, the edges in unorganized point cloud are jagged. The outline extraction from a planar roof points is the process of generalization. Usually, after 3D model reconstruction, an evaluation is necessary to check which accuracy level has been achieved when compared to the original point cloud. Therefore, when using 3D models for shadow matching, there are several issues that needed to be considered: (i) what kind of data sources has been used for 3D model generation; (ii) what kind of method has been used for the 3D model reconstruction; (iii) what kind of accuracy level has been achieved in the resulting 3D models. This paper will not extend these topics too far because they are beyond the scope of this paper. However, it is clear that when the 3D models are reconstructed, the loss of the accuracy is inevitable.

The point cloud data quality of the adopted MLS platform was already analysed in Kaartinen et al. [14], there was no need to analyse further the point cloud quality and better than 5 cm accuracy of the data is guaranteed in the whole experiment, which is more accurate than most available 3D models. A potential application for such technology is that it can utilize the point cloud generated by the LiDAR sensor equipped by autonomous driving car in near future.

In this research, we investigate the feasibility of utilizing point cloud collecting by MLS platform to generate the shadow mask by removing the objects which only attenuating the GNSS signal. The enhancement comparison between the proposed method and other will be discussed future, if a centimetre level accurate 3D model of the investigated area

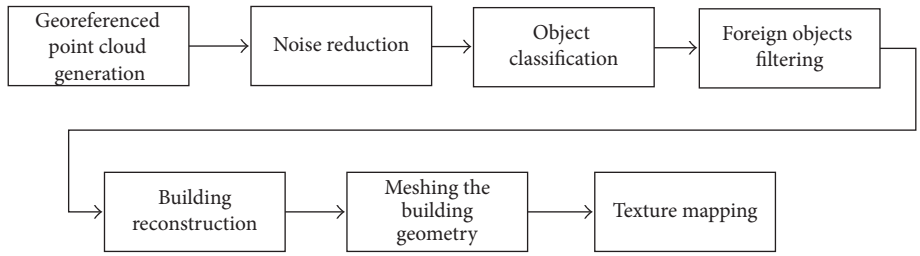



Figure 1: General processing chain from MLS data collection to 3D city model.

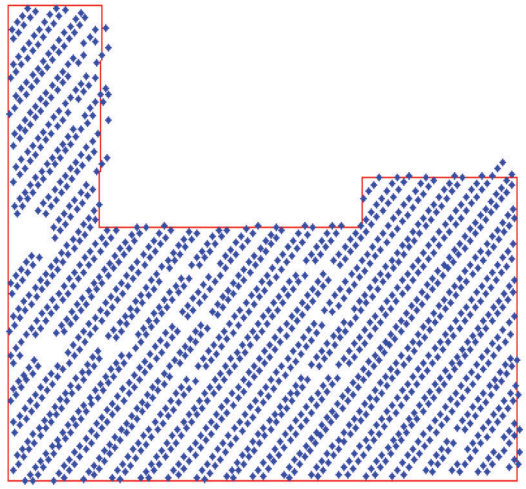


Figure 2: An example of outline extraction from planar roof points: a part of 3D model reconstruction.

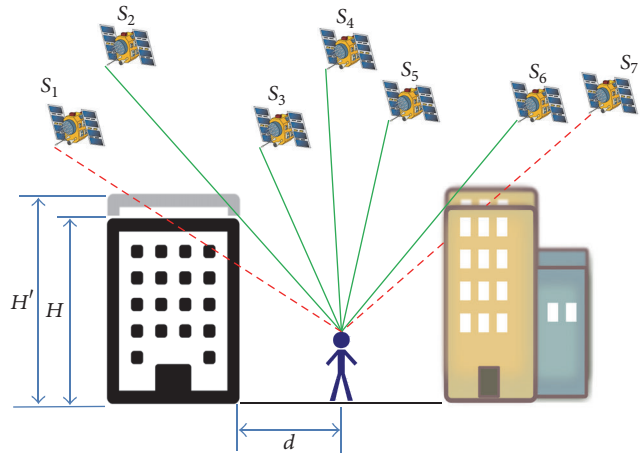



Figure 3: How the accuracy of the model affects the shadow matching results.

is available. Besides, the performance enhancement is a comprehensive result affected by geometry of GNSS constellation, selected GNSS receivers, and the adopted algorithm (such as particle filter and optimized visibility scoring scheme [4]).

## 3. Methodology and Algorithm

As Figure 3 illustrating a typical shadow matching scenario in the city canyon, assuming the distance between the pedestrian and the building is $d$, the height of the building in 3D model is $H'$ against its truth $H$; the error $E$ introduced by the inaccuracy of the model $e$ (normally in several decimetres to meters level) can be calculated with

$$\begin{aligned} E &= \operatorname{arctg}\left(\frac{d}{H}\right) - \operatorname{arctg}\left(\frac{d}{H'}\right) \\ &= \operatorname{arctg}\left(\frac{d}{H}\right) - \operatorname{arctg}\left(\frac{d}{H+e}\right). \end{aligned} \tag{1}$$

As a result, satellites $S_2$ and $S_6$ are excluded for the positioning computation, when they actually should be used. This is because these satellites are actually in view, but the inaccuracy in the 3D model will excluded satellites $S_2$ and $S_6$ from the observed list. It implies that the less accurate 3D model might cause the shadow matching positioning not applicable in practical. It can also be concluded that this research has potential benefits to improve the positioning accuracy with shadow matching method if a more accurate 3D city model is available.

One issue that has been addressed is that, in this research, we do not utilize the existing 3D model but investigate the method to generate the GNSS shadow mask directly from 3D point cloud collected from a MLS system. 3D model is actually a simplified expression of physical environment which ignores some details on purpose, while the point cloud collected by the MLS contains the most detailed environmental object. Thus the methodology investigated in this research is to filter out the unnecessary information to generate a GNSS shadow mask as precise as possible, by considering the physical natures of the scanned environmental objects without generating a real 3D model from the point cloud.

In this research, a small-footprint phase-based laser scanner is utilized. The major reasons that phase-based model is used rather than a pulse-based version are explained in the following.

(1) Phase-based model has smaller field of view (FOV); the one utilized for this research has 0.19 mRad FOV while most pulse-based laser scanners are with several mRad FOV. Considering the 25 meters' range, the beam diameter at exit is 2.25 mm and the maximum footprint size of the adopted phased based model is

7.3 mm, which is considerably smaller in comparison to 10–20 cm of larger FOV pulsed based scanners. To generate a point cloud as detailed as possible, the footprint should be small due to the fractional interception of laser energy by scattered objects within the laser footprint. Large footprint laser scanner will introduce more measurement error [34–38].

(2) Phase-based model has higher acquisition speed, comparing with pulsed based model; the acquisition speed of the selected model can be approximately 1 M point per second against 10~100 k of most pulse-based laser scanners. It implies that a considerable denser point cloud can be generated with a higher efficient manner, comparing with that generated by the pulse-based model. In other words, more details can be unveiled by the denser point cloud.

(3) Phase-based model has higher range resolution. In this research the range resolution is 1 mm against one to several centimetres' range resolution of most pulsed based models.

(4) Phase-based model adopts the laser with less transmission power comparing with pulsed based laser scanner. Eye safety is an issue determining the acceptance of laser scanners especially for massive civil applications.

However, the field measurements indicate that the phase-based laser scanner is more sensitive to environment factor and results in higher noise levels [38]. A noise mitigating procedure is necessary to filter out the noise measurement before calculating the shadow mask.

The data sources conducted in this research are MLS point cloud (from a FARO Focus laser scanner), trajectory data (from a NovAtel SPAN georeferenced system), GNSS data collected by a dual-frequency receiver (a NovAtel OEMV receiver), and precise satellite ephemeris downloaded from an Internet service.

MLS point cloud is a set of georeferenced points, which contain 3D coordinates $(X, Y, Z)$ and intensity values. The research adopts ETRS-TM35FIN with GRS80 ellipsoidal height map coordinate system, in which $X$ is pointing to the east, $Y$ towards the north, and $z$-axis upwards. In this research, when we mention "the top view," it means in $XY$ plane view.

Usually, laser scanning point cloud contains much noise as a result of failed ranging or multiple reflections when reflecting surface is smaller than the footprint of the scanning laser point, which will directly lead to wrong results. Therefore, it is comparably important to filter out the noise, and the noise reduction is the foundation of all point cloud processing. Otherwise such sparsely existing spatial noise might be recognized as the highest obstruction point in further processing. Normally the noise is present as isolated point(s) and can be detected efficiently with a spatial filter. An adaptive spatial filter (ASF) was utilized to mitigate the noise from raw laser scanning points. After noise was filtered out, we designed another ASF for detecting points coming from objects that do not interfere with GNSS, for example, the hanging power cable, flag. From a GNSS user perspective, those objects cast little GNSS shadow on the antenna of a receiver; thus, such objects do not attenuate the physical signal strength of the GNSS. However, such objects do influence the calculation of the highest obstruction angles of the scene. The ASF can also determine the objects which only attenuate the GNSS signal rather than block it, such as plantings. Next, the highest obstruction angle for each azimuth along a grid point can be calculated. Finally, the GNSS visibility map for the whole area is built with a high density two-dimensional grid, for example, with 1-meter spacing.

*3.1. Noise Mitigation Processing of Raw Laser Point Cloud.* The ASF for noise mitigation was developed based on the isolated distribution characteristic of the noise with the following steps.

(1) The point cloud is projected into 2D views of $XY$, $XZ$, and $YZ$ planes.

(2) From each 2D view, we use a bivariate histogram to analyse each grid (with a grid size of $15 \times 15$ meters) and calculate the number of points that fall in each grid.

(3) If the number of points is less than the threshold of the filter, (150 as initialized value with current system configuration), it was considered as noise.

Because the filter is mainly for mitigating spatial noise, we refer to it as a "spatial filter." The threshold setting of the spatial filter adaptively relies on the density of the point cloud, noise distribution, the size of grid, and the scene situation. For example, when a scene contains water area or a large area of glass-made objects, noise level will be higher. Therefore, user knowledge and experience are needed for setting the threshold value.

*3.2. Space Hanging Objects and Plantings Detection and Removal.* Space hanging objects, for example, pipelines, powerlines, and hanging flags are ubiquitous and usually as a part of a city's infrastructure especially in the city canyon, which cast too little GNSS shadow on the GNSS receiver, to be considered changing the visibility of the satellites. Therefore, these objects need to be removed from the scene before the calculation of the highest obstruction angle. In addition, considering the complex of the scene, for example, when a person is located under a bridge or a tunnel, the concrete structure of these objects would block the signals. In this case, the high buildings near the bridge cannot always be considered as the highest obstruction angles. Therefore, a voxel based algorithm is developed to detect the location of a view point: in open area or under a bridge or a tunnel. We assume a voxel with a size of $5 \times 5 \times 5$ meters. We put this voxel centred at the view point and the bottom from 2 meters height above ground. The points inside the voxel can be analysed according to the number and the density of the points. When the number and the density of the points are greater than the thresholds, it is accepted as the bridge or tunnel points.

Whole object (bridge or tunnel) can be detected by applying region grow method. We analyse the neighbouring points of each candidate for the highest point. If the number of points around the candidate has an ignorable change in vertical direction, we consider it as an object hanging in the air and it is therefore removed. Poles and wires can also be removed by other specific algorithms, such as [25] for poles and [33] for wires.

The highest points generated by the plantings have its unique sparse pattern against the linear sky line generated by regular buildings when we project it into the top view as Figure 12(c) presents. It can be easily detected and removed with a two-dimensional spatial filter.

*3.3. The Highest Point Extraction and the Highest Obstruction Angle Calculation.* After the noise and space hanging object detection and removal, for each grid point, the GNSS shadow map of the scene can be evaluated by choosing the point cloud within a radius of 25 meters 2D-distance from the grid point. A low building close to the user might yield to larger elevation angle than a high building far away. 25 meters is a compromise and empirical value between the computational complexity and performance and the total number of point cloud within the radius is approximate hundreds of million based on current system setup. In a city canyon, the heights of buildings vary from 5 meters to over 100 meters. The threshold for the 3D distance from grid point is usually difficult to be defined. Therefore, it is more accurate to define a scene by choosing a 2D distance from a grid point.

We set each grid point as Grid_$P$ whereas the scene points with 25 meters or less 2D distance from Grid_$P$ are designated as Scene_$P$. A grid point Grid_$P$ stands for a pedestrian standing in this position whereas the range of the scene points represents a visual area in the position of Grid_$P$. Grid_$P$ is calculated based on trajectory data collected by NovAtel SPAN, and about 1.4 meters from ground surface. Figure 4 gives an example of the relationship between a grid point and its visual area in the 3D view and in the top view. In order to demonstrate the obstruction of a scene in terms of the highest points, from a top view of the Scene_$P$, the Scene_$P$ is divided into 360 sectors from the grid point at Grid_$P$. Each sector is 1°.

After the highest points of the surrounding scene of each grid point have been obtained, elevation angle of the highest point with respect to the vertical direction is calculated. Figure 5 shows the definitions of the vertical angle. The red dot is the grid point while the green dot shows one highest point surrounding in a scene. Grid_$P_i$ is the $i$th grid point. Scene_$P_{ij}$ presents the highest point of $j$th degree of azimuth of the surrounding scene of the $i$th grid point. $\alpha$ represents the elevation angle of the point "Scene_$P_{ij}$" with respect to the vertical direction.

The angles are calculated as follows:

$$\alpha = \arcsin \frac{Z_{\text{scene\_}P_{ij}} - Z_{\text{Grid\_}P_i}}{\rho}, \tag{2}$$

where "$Z_{\text{scene\_}P_{ij}}$" and "$Z_{\text{Grid\_}P_i}$" are the $z$-axis coordinates of the point "Scene_$P_{ij}$" and the point "Grid_$P_i$," respectively; "$\rho$" stands for the 3D distance between the point "Scene_$P_{ij}$" and the point "Grid_$P_j$".

*3.4. The GNSS Shadow Mask.* Based on the elevation angle of each azimuth, the sky plot of GNSS shadow mask of each grid can be drawn. Figure 6 presents one example of azimuth-elevation pairs in a sky plot in a test field, where the red circles stand for Space Vehicle's (SV)/GNSS satellites' positions based on the ephemeris data and with corresponding satellite ID the blue line stands for the boundary of the GNSS shadow extracted from the MLS data with the above-mentioned ASF applied. GNSS satellites with elevation angles below the determined boundary were blocked by the buildings from the GNSS users' perspective. Therefore, the GNSS shadow mask of each grid extracted from the MLS became a useful information for positioning. Such spatial information could be utilized by the GNSS shadow matching methodology to enhance the positioning accuracy.

*3.5. Two Scenarios for Investigation.* In this research, two special scenarios of the city canyon are investigated besides typical scenario: the first is scenario with hanging objects such as flags and cables as Figure 7(a) presents and Figure 7(b) presents the city canyon with plantings which might attenuate the GNSS RF signal but not block it. It is well known that the GNSS observability along the canyon direction is better than the cross direction due to the topography of the city canyon. However, if there are some hanging objects like power cables, hanging lights, and flags that exist, it might be processed as the highest obstruction point in the along canyon direction which might redefine, in most cases narrow down, the boundary of the GNSS shadow mask. Thus an adaptive spatial filter needs to be designed to filter out the hanging objects in the point cloud. We name such hanging objects cases as "CASE I" in the following context. The same consequence occurs if there are some plantings existing nearby as Figure 7(b) presents. When the plantings are close to the observing point, the highest obstruction point introduced from the point cloud of the plantings might also redescribe the GNSS shadow mask. Thus another ASF should be designed to deal with the case. Such scenario is named as "CASE II" in the following research.

## 4. MLS Data Collection

MLS data used for the experiment were collected by ROAMER mobile mapping system developed by the Finnish Geospatial Research Institute (FGI) as Figure 8 presents, in Tapiola shopping centre area, Espoo, Finland, which is a 300-by-300-meter area with buildings of varying size and height [39]. The MLS system can be applied to different platforms, that is, car, trolley, and boat and also for personal backpack [40]. The effective range of ROAMER was 78 meters and the scanning rate could reach approximate 1 M/s. With regard to the experiment, the system employed a trolley as a platform to collect data to enter also pedestrian alleys at speed of about 1 m/s.

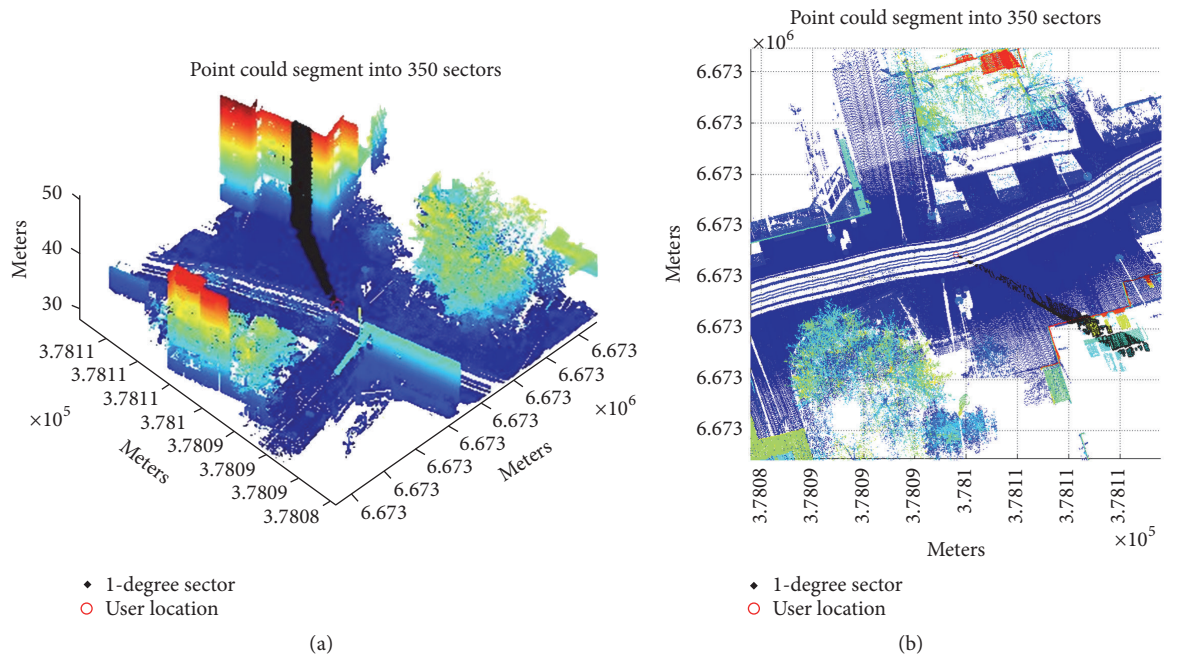



FIGURE 4: Illustration of a visual area at a grid point (Grid_$P$) and how the points are divided into 360 sectors (1° as one sector) in (a) 3D view and (b) $x$-$y$ plane view.

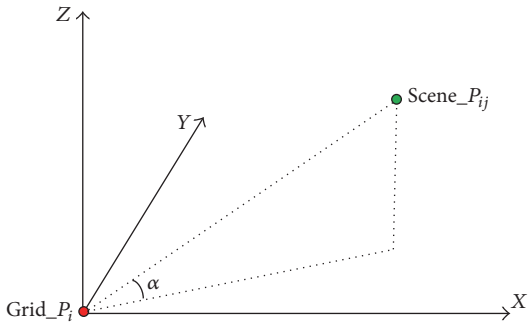



FIGURE 5: The vertical angle calculation.

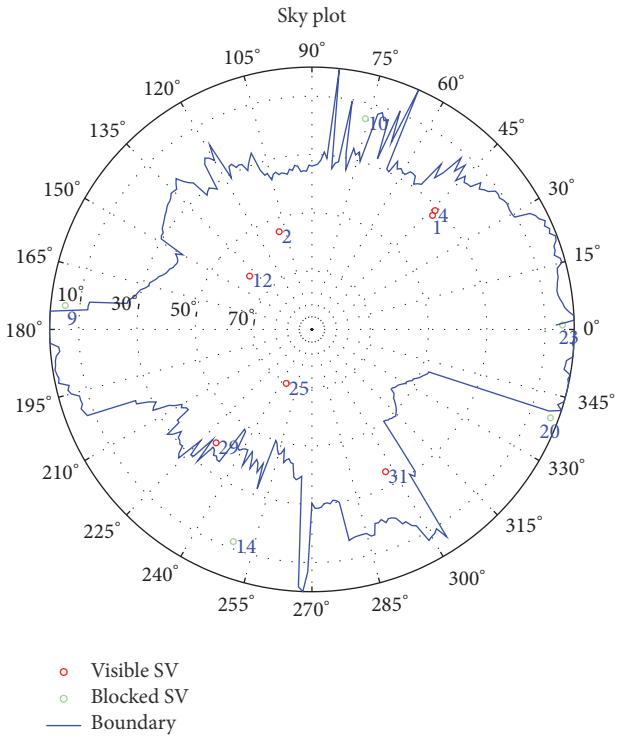



FIGURE 6: GNSS shadow mask sky plot.

## 5. Results and Discussions

Figure 9 illustrates the effectiveness of the ASF for noise mitigation. The red dots present the noise measurements within the MLS collected point cloud. Most of the sparse noise hanging in space has been filtered out with the filter. The resulting filtered point cloud was used for the highest point extraction.

Figure 10 shows the detection of the highest points by removing the cable-like objects. The black dots are the highest points of each azimuth in one scene, and the red point in the figure stands for a grid point. The colours in the figure indicate height. From the figure, it can be observed that all space hanging objects in the scene are detected and removed.

Figure 11 shows a comparison of the highest point extraction with Figure 11(a) and without Figure 11(b) the space hanging objects. Without adapting the spatial filter, many of the extracted mask points are located on three hanging cables between two buildings in the scene. Cables are detected and deleted with the ASF from the highest points, list as Figure 11(b) shows. The difference is more obvious when we project all highest point to $XY$ plane,

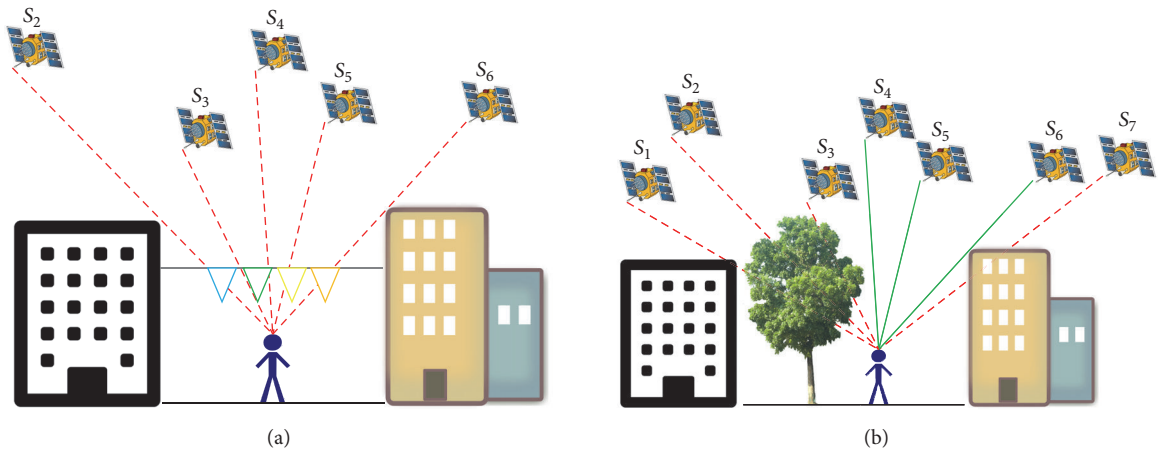



Figure 7: Two special scenarios investigated in this research: (a) city canyon with hanging objects; (b) city canyon with nearby plantings.

Figure 8: ROAMER on data collecting in Tapiola, Finland.

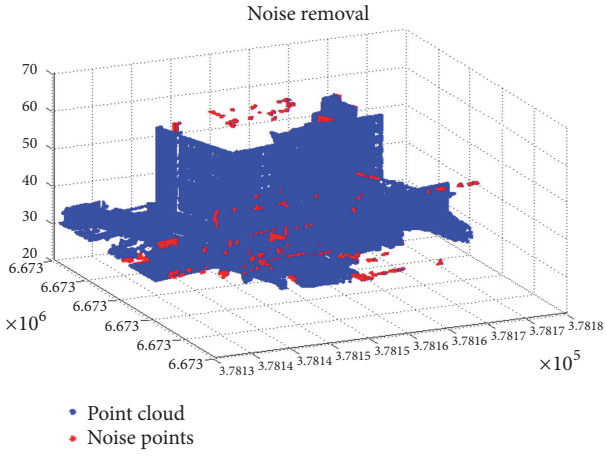



Figure 9: Noise mitigation results.

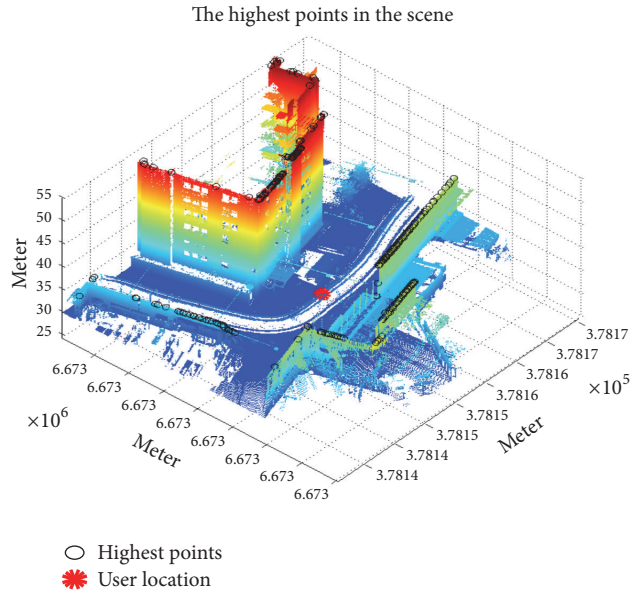



Figure 10: The results of the highest points detection after the space hanging object filter has been applied.

as Figures 11(c) and 11(d) present. Figure 11(e) shows the difference of the two GNSS shadow masks. The blue circle line depicts the boundary of shadow with the spatial filter, which has much lower elevation angle in the cross canyon (north-south) direction in comparison to the red star line which is drawn based on the unfiltered data. From the red star line, we can perceive that there are three cables existing in the observed area and there are two dents in the red circle between 50° to 55° and −135° to −140°; it is identified to be two lights hung on power cables for illustration in Tapiola. We can observe that the design ASF can correctly filter out the hanging objects and generate the GNSS shadow mask which is more precise in depicting the GNSS visibility in city canyon environments for CASE I scenario.

Figure 12 presents the effectiveness of the design ASF for CASE II scenario in city canyon. As Figures 12(a) and 12(c) shown, all sparse distributed highest points introduced by the plantings have been filtered out and a more clear GNSS mask is generated in Figures 12(b) and 12(d). By comparing the GNSS shadow masks generated with the method with and without ASF, some conclusions can be drawn: (1) the proposed ASF can correctly filter out the highest point generated by the nearby plantings; (2) the signal attenuated

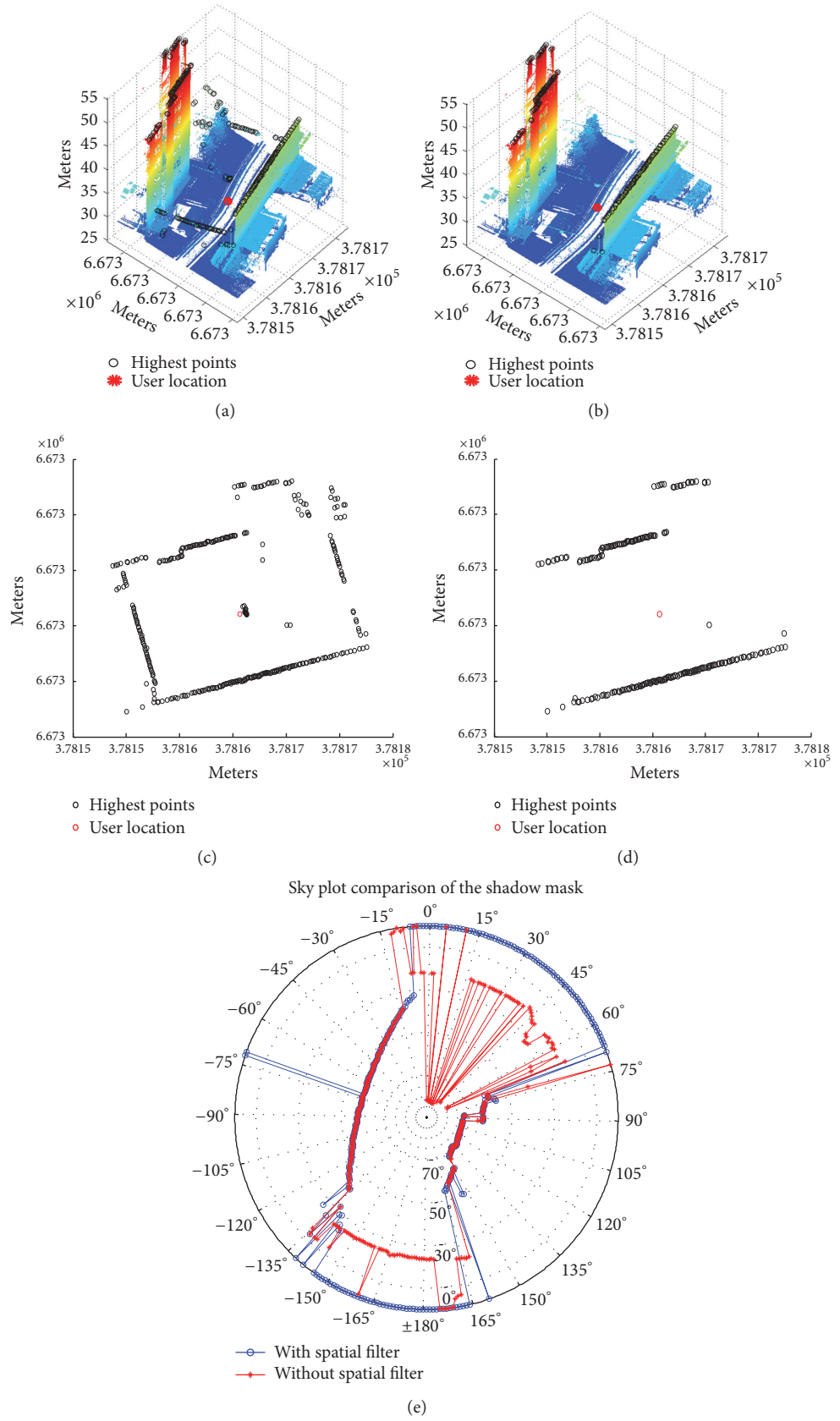



FIGURE 11: (a) Highest points extraction without ASF, (b) highest points extraction with ASF, (c) top view of the extracted highest points in $XY$ plane without ASF, (d) top view of the extracted highest points in $XY$ plane with ASF GNSS, and (e) shadow mask sky plot generated by the methods with and without ASF in CASE I.

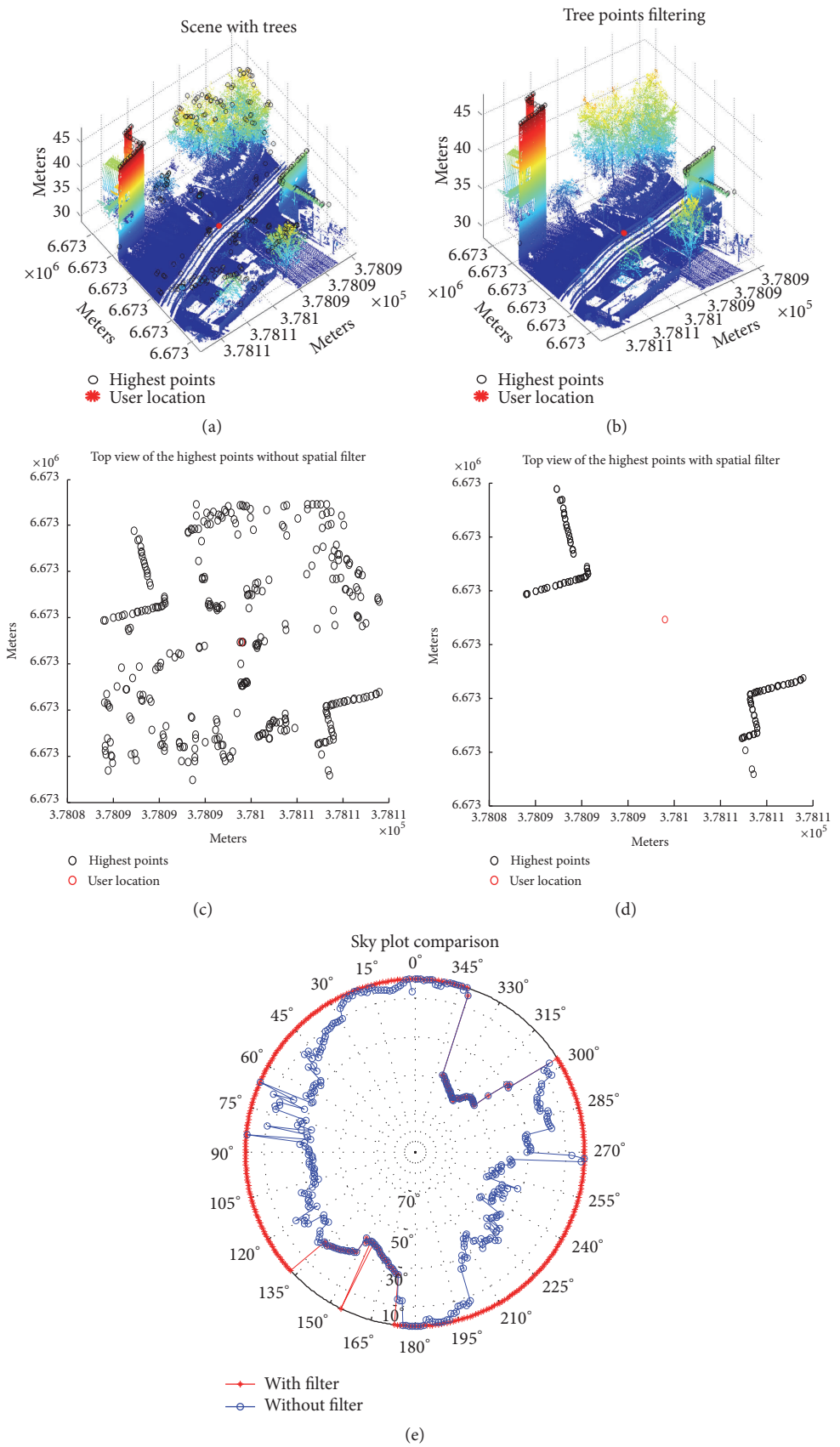



FIGURE 12: (a) Highest points extraction without ASF, (b) highest points extraction with ASF, (c) top view of the extracted highest points in $XY$ plane without ASF, (d) top view of the extracted highest points in $XY$ plane with ASF GNSS, and (e) shadow mask sky plots generated by the methods with and without ASF in CASE II.

by the planting can also be utilized as a special signal of opportunity (SOP) for positioning to improve the location accuracy: because the attenuation introduced by trunk or foliage should be lower comparing the attenuation caused by buildings resulting in lower SNR GNSS signal.

## 6. Conclusion

By analysing the MLS point cloud data and by applying developed ASF, it is concluded that MLS data can be used to extract GNSS shadow masks after a series of appropriate processing steps to filter out the hanging objects and the plantings. Since the seamless MLS point cloud data has centimetre accuracy, the extracted GNSS shadow mask is more precise, compared with the ones generated from traditional 3D models, where the model error is several decimetres [1] or even higher. Since point cloud from city and roadside areas are collected by large geospatial data providers with mobile mapping and laser scanning technology, such as Nokia HERE and Google, who are also the major players in the fields of positioning and navigation, there are possibilities to implement the presented methods for the benefit of GNSS users in city environments.

In future research, we will generate a dense two-dimensional grid with 1-meter spacing GNSS shadow mask database for testing GNSS shadow matching to enhance the position accuracy in city canyons. As far as we know, it would be the most detailed and dense database with centimetre level accuracy. More specific field tests are planned to be conducted in the test area to evaluate its improvement of pedestrian navigation applications by comparing the performance between the MLS method and the 3D city models for the elevation mask determination. We also aim to analyse the attenuation introduced by trunk and foliage on GNSS signals to investigate a more precise GNSS shadow matching algorithm in city canyon environments that takes into account city plants.

## Competing Interests

The authors declare that they have no competing interests.

## Acknowledgments

The research is financially supported by Academy of Finland (New Laser and Spectral Field Methods for In Situ Mining and Raw Material Investigations (292648)) and Strategic Research Council at the Academy of Finland is acknowledged for financial support (Project Decision no. 293389). Additionally, Chinese Academy of Science (181811KYSB20130003, 181811KYSB20160113), China Ministry of Science and Technology (2015DFA70930), and National Natural Science Foundation of China (41304004) are acknowledged.

## References

[1] L. Wang, P. D. Groves, and M. K. Ziebart, "Urban positioning on a smartphone: real-time shadow matching using GNSS and 3D city models," *Inside GNSS Magazine*, vol. 8, no. 6, pp. 44–56, 2013.

[2] C. Tiberius and E. Verbree, "GNSS positioning accuracy and availability within Location Based Services: the advantages of combined GPS-Galileo positioning," in *Proceedings of the ESA/Estec Workshop on Satellite Navigation User Equipment Technologies*, G. S. Granados, Ed., pp. 1–12, ESA Publications Division, Noordwijk, The Netherland, 2004.

[3] P. D. Groves, "Shadow matching: a new GNSS positioning technique for urban canyons," *Journal of Navigation*, vol. 64, no. 3, pp. 417–430, 2011.

[4] L. Wang, P. D. Groves, and M. K. Ziebart, "GNSS shadow matching: improving urban positioning accuracy using a 3D city model with optimized visibility prediction scoring," in *Proceedings of 25th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS '12)*, pp. 423–437, Nashville, Tenn, USA, 2012.

[5] L. Wang, P. D. Groves, and M. K. Ziebart, "Multi-constellation GNSS performance evaluation for urban canyons using large virtual reality city models," *Journal of Navigation*, vol. 65, no. 3, pp. 459–476, 2012.

[6] M. Adjrad and P. D. Groves, "Intelligent urban positioning using shadow matching and GNSS ranging aided by 3D mapping," in *Proceedings of the 29th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS '16)*, Portland, Ore, USA, September 2016.

[7] M. Adjrad and P. D. Groves, "Enhancing conventional GNSS positioning with 3D mapping without accurate prior knowledge," in *Proceedings of the 28th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS '15)*, pp. 2397–2409, Tampa, Fla, USA, September 2015.

[8] R. Chen and J. Liu, "Mitigating GNSS positioning errors using 3D spatial attributes of map data," Finnish Patent: 20110073, January 2011.

[9] J. T. Isaacs, A. T. Irish, F. Quitin, U. Madhow, and J. P. Hespanha, "Bayesian localization and mapping using GNSS SNR measurements," in *Proceedings of the IEEE/ION Position, Location and Navigation Symposium (PLANS '14)*, pp. 445–451, IEEE, Monterey, Calif, USA, May 2014.

[10] J. Bradbury, "Prediction of urban GNSS availability and signal degradation using virtual reality city models," in *Proceedings of the 20th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS '07)*, pp. 2696–2706, Fort Worth, Tex, USA, September 2007.

[11] J. T. Isaacs, A. T. Irish, F. Quitin, U. Madhow, and J. P. Hespanha, "Bayesian localization and mapping using GNSS SNR measurements," in *Proceedings of the IEEE/ION Position, Location and Navigation Symposium (PLANS '14)*, pp. 445–451, May 2014.

[12] R. Kumar and M. G. Petovello, "A novel GNSS positioning technique for improved accuracy in urban canyon scenarios using 3D city model," in *Proceedings of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS '14)*, vol. 812, pp. 2139–2148, Tampa, Fla, USA, 2014.

[13] J.-Y. Han and P.-H. Li, "Utilizing 3-D topographical information for the quality assessment of a satellite surveying," *Applied Geomatics*, vol. 2, no. 1, pp. 21–32, 2010.

[14] H. Kaartinen, J. Hyyppä, A. Kukko, A. Jaakkola, and H. Hyyppä, "Benchmarking the performance of mobile laser scanning systems using a permanent test field," *Sensors*, vol. 12, no. 9, pp. 12814–12835, 2012.

[15] M. Rutzinger, B. Höfle, S. O. Elberink, and G. Vosselman, "Feasibility of facade footprint extraction from mobile laser

scanning data," *Photogrammetrie, Fernerkundung, Geoinformation*, vol. 2011, no. 3, pp. 97–107, 2011.

[16] C. Frueh, S. Jain, and A. Zakhor, "Data processing algorithms for generating textured 3D building facade meshes from laser scans and camera images," *International Journal of Computer Vision*, vol. 61, no. 2, pp. 159–184, 2005.

[17] L. Zhu, J. Hyyppä, A. Kukko, H. Kaartinen, and R. Chen, "Photorealistic building reconstruction from mobile laser scanning data," *Remote Sensing*, vol. 3, no. 7, pp. 1406–1426, 2011.

[18] H. Zhao and R. Shibasaki, "Reconstructing a textured CAD model of an urban environment using vehicle-borne laser range scanners and line cameras," *Machine Vision and Applications*, vol. 14, no. 1, pp. 35–41, 2003.

[19] D. Manandhar and R. Shibasaki, "Auto-extraction of urban features from vehicle-borne laser data," *International Archives of Photogrammetry, Remote Sensing and Spatial, Information Sciences*, vol. 34, pp. 433–438, 2002.

[20] M. Rutzinger, A. K. Pratihast, S. J. Oude Elberink, and G. Vosselman, "Tree modelling from mobile laser scanning datasets," *Photogrammetric Record*, vol. 26, no. 135, pp. 361–372, 2011.

[21] S. Pu, M. Rutzinger, G. Vosselman, and S. Oude Elberink, "Recognizing basic structures from mobile laser scanning data for road inventory studies," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 66, no. 6, pp. S28–S39, 2011.

[22] E. Puttonen, A. Jaakkola, P. Litkey, and J. Hyyppä, "Tree classification with fused mobile laser scanning and hyperspectral data," *Sensors*, vol. 11, no. 5, pp. 5158–5182, 2011.

[23] A. Jaakkola, J. Hyyppä, A. Kukko et al., "A low-cost multi-sensoral mobile mapping system and its feasibility for tree measurements," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 65, no. 6, pp. 514–522, 2010.

[24] C. Brenner, "Extraction of Features from mobile laser scanning data for future driver assistance systems," in *Advances in GIScience-Proceedings of the 12th AGILE Conference*, Lecture Notes in Geoinformation and Cartography, pp. 25–42, Springer, Berlin, Germany, 2009.

[25] M. Lehtomäki, A. Jaakkola, J. Hyyppä, A. Kukko, and H. Kaartinen, "Detection of vertical pole-like objects in a road environment using vehicle-based laser scanning data," *Remote Sensing*, vol. 2, no. 3, pp. 641–664, 2010.

[26] M. Lehtomäki, A. Jaakkola, J. Hyyppä, A. Kukko, and H. Kaartinen, "Performance analysis of a pole and tree trunk Detection method for mobile laser scanning data," in *Proceedings of the ISPRS Calgary Workshop on Laser Scanning*, pp. 197–202, ISPRS, Calgary, Canada, August 2011.

[27] A. Golovinskiy, V. G. Kim, and T. Funkhouser, "Shape-based recognition of 3D point clouds in urban environments," in *Proceedings of the 12th International Conference on Computer Vision (ICCV '09)*, pp. 2154–2161, October 2009.

[28] H. Yokoyama, H. Date, S. Kanai, and H. Takeda, "Pole-like objects recognition from mobile laser scanning data using smoothing and principal component analysis," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 38, pp. 115–120, 2011.

[29] A. Jaakkola, J. Hyyppä, H. Hyyppä, and A. Kukko, "Retrieval algorithms for road surface modelling using laser-based mobile mapping," *Sensors*, vol. 8, no. 9, pp. 5238–5249, 2008.

[30] S.-J. Yu, S. R. Sukumar, A. F. Koschan, D. L. Page, and M. A. Abidi, "3D reconstruction of road surfaces using an integrated multi-sensory approach," *Optics and Lasers in Engineering*, vol. 45, no. 7, pp. 808–818, 2007.

[31] C. McElhinney, P. Kumar, C. Cahalane, and T. McCarthy, "Initial results from European Road Safety Inspection (EuRSI) mobile mapping project," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 38, pp. 440–445, 2010.

[32] M. Vaaja, J. Hyyppä, A. Kukko, H. Kaartinen, H. Hyyppä, and P. Alho, "Mapping topography changes and elevation accuracies using a mobile laser scanner," *Remote Sensing*, vol. 3, no. 3, pp. 587–600, 2011.

[33] P. Axelsson, "Processing of laser scanner data—algorithms and applications," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 54, no. 2-3, pp. 138–147, 1999.

[34] J. Jutila, K. Kannas, and A. Visala, "Tree measurement in forest by 2D laser scanning," in *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA '07)*, pp. 491–496, Jacksonville, Fla, USA, June 2007.

[35] J. Tang, Y. Chen, L. Chen et al., "Fast fingerprint database maintenance for indoor positioning based on UGV SLAM," *Sensors*, vol. 15, no. 3, pp. 5311–5330, 2015.

[36] Y. Chen, J. Tang, J. Hyyppä et al., "Automated stem mapping using SLAM technology for plot-wise forest inventory," in *Proceedings of the Ubiquitous Positioning Indoor Navigation and Location-Based Services (UPINLBS '14)*, pp. 130–134, Corpus Christi, Tex, USA, November 2014.

[37] O. Ringdahl, P. Hohnloser, T. Hellström, J. Holmgren, and O. Lindroos, "Enhanced algorithms for estimating tree trunk diameter using 2D laser scanner," *Remote Sensing*, vol. 5, no. 10, pp. 4839–4856, 2013.

[38] T. Nuttens, C. Stal, J. Wisbecq, G. Deruyter, and A. De Wulf, "Field comparison of pulse-based and phase-based laser scanners for civil engineering applications," in *Proceedings of the 14th International Multidisciplinary Scientific Geoconference and EXPO (SGEM '14)*, pp. 169–176, Sofia, Bulgaria, June 2014.

[39] A. Kukko, C.-O. Andrei, V.-M. Salminen et al., "Road environment mapping system of the Finnish Geodetic Institute—FGI ROAMER," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, no. 3/W52, pp. 241–247, 2007.

[40] X. Liang, A. Kukko, H. Kaartinen et al., "Possibilities of a personal laser scanning system for forest mapping and ecosystem services," *Sensors*, vol. 14, no. 1, pp. 1228–1248, 2014.

Hindawi
Mobile Information Systems
Volume 2017, Article ID 8294248, 20 pages
https://doi.org/10.1155/2017/8294248

*Research Article*

# An Analysis of Impact Factors for Positioning Performance in WLAN Fingerprinting Systems Using Ishikawa Diagrams and a Simulation Platform

**Keqiang Liu, Yunjia Wang, Lixin Lin, and Guoliang Chen**

*School of Environmental Science and Spatial Informatics, China University of Mining and Technology, 1 Daxue Road, Xuzhou City, Jiangsu Province 221116, China*

Correspondence should be addressed to Yunjia Wang; wycgcumt@163.com

Received 31 October 2016; Revised 10 January 2017; Accepted 24 January 2017; Published 21 February 2017

Academic Editor: Liang Chen



Many factors influence the positioning performance in WLAN RSSI fingerprinting systems, and summary of these factors is an important but challenging job. Moreover, impact analysis on nonalgorithm factors is significant to system application and quality control but little research has been conducted. This paper analyzes and summarizes the potential impact factors by using an Ishikawa diagram considering radio signal transmitting, propagating, receiving, and processing. A simulation platform was developed to facilitate the analysis experiment, and the paper classifies the potential factors into controllable, uncontrollable, nuisance, and held-constant factors considering simulation feasibility. It takes five nonalgorithm controllable factors including APs density, APs distribution, radio signal propagating attenuation factor, radio signal propagating noise, and RPs density into consideration and adopted the OFAT analysis method in experiment. The positioning result was achieved by using the deterministic and probabilistic algorithms, and the error was presented by RMSE and CDF. The results indicate that the high APs density, signal propagating attenuation factor, and RPs density, with the low signal propagating noise level, are favorable to better performance, while APs distribution has no particular impact pattern on the positioning error. Overall, this paper has made great potential contribution to the quality control of WLAN fingerprinting solutions.

## 1. Introduction

During the past two decades, there has been an exceptional development in localization and positioning field benefiting from global positioning system (GPS) [1]. However, GPS performs weakly when the radio signal is obstructed, for example, in urban canyons and indoor environments. To fill the gap, many indoor positioning systems have been proposed using different methods [2–5], including triangulation, proximity, scene analysis, and pedestrian dead reckoning (PDR). There are already many commercial off-the-shelf (COTS) systems for indoor localization, such as Situm (https://situm.es/en), Kio-RTLS (http://www.eliko.ee), OpenRTLS (https://openrtls.com), Aero Scout (http://www.aeroscout.com), and Ubisense (http://ubisense.net/en). Among these different indoor localization systems, location fingerprinting is a solution using scene analysis method and usually working with wireless local area network (WLAN) received signal strength indicator (RSSI), for example, the Skyhook (http://www.skyhook-wireless.com/) commercial positioning system. The scene analysis algorithm consists of two phases: offline learning and online positioning. Firstly, it builds a location fingerprint database on some reference points (RPs) with given location coordinates during the offline learning phase. Then in the online positioning phase, the main work is to search the nearest RP or RPs by using RSSI received on an unknown point. There are two kinds of common searching algorithms in use: the deterministic [6] and probabilistic [7, 8] algorithms.

In a WLAN location fingerprinting system, there are many factors which may have impact on positioning performance. For example, location fingerprinting is a cost-saving solution by using previously deployed access points

(APs) for communication. However, these nondedicated APs also have limitations for localization. For example, only a limit number of APs can be detected in a particular environment. Moreover, the RSSI of WLAN is unstationary during radio wave attenuating from APs to RPs under the influence of environment noise. Therefore, the number of APs and unstationary RSSI may influence the positioning result. Search and summary of the potential impact factors are thus important to guiding the further impact analysis, yet a challenging task, which has not been paid much attention.

There have been many studies on WLAN location fingerprinting in different aspects, such as developing diverse systems, implementing with novel searching algorithms, and constructing fingerprint database using different approaches, which will be reviewed in Section 2 in detail. However, little research effort has been made on impact factors due to some limitations in conducting analysis experiments. Comparing with factors such as searching algorithm and parameters in an algorithm, it is less convenient to conduct experiments for analyzing some nonalgorithm factors. But, nonalgorithm factors are also important in technique application and system quality control. For instance, environment noises cause the instability of WLAN RSSI, which may have influences on positioning results. However, some environment factors such as the crowd, temperature, and humidity are not easy to control in a real experiment field. Moreover, the experiment will be complicated, and the number of tests will be extremely large when interactions of all factors are taken into account. For example, assuming there are $k$ factors that are regarded as factors of interest and the $i$th factor has $n_i$ levels to be analyzed, the number of tests would be $\prod_i^k n_i$ in a factorial experiment.

To address these issues, the research objectives of this study are as follows:

(1) To summarize the potential factors and provide a reference for factors research, this paper uses an Ishikawa diagram [9] method with consideration of WLAN radio signal transmitting, propagating, receiving, and processing in WLAN location fingerprinting systems (Section 3).

(2) To facilitate the analysis experiment of factors and impacts, an open-source simulation WLAN location fingerprinting platform (https://github.com/cumtlkq/WiFiPosSimu) was developed, and this paper classifies the factors into four types including controllable, uncontrollable, nuisance, and held-constant factors considering simulation feasibility. All the categorized factors are presented in another Ishikawa diagram (Section 4).

(3) To promote the application and quality control of WLAN location fingerprinting, this paper selected five nonalgorithm controllable factors (APs density, APs distribution, radio signal propagating attenuation factor, radio signal noise, and RPs density) as factors of interest (Section 4).

(4) To reduce the complexity and number of tests in a factorial experiment, the analysis experiment was conducted using the one-factor-at-a-time (OFAT) method with different test settings to analyze the impact patterns of the selected factors on positioning error (Section 5).

## 2. Related Works

The most famous fingerprint localization system was proposed by Bahl and Padmanabhan [6], named RADAR. It combined location fingerprinting with triangulation on signal propagation modeling to determine user location. The median resolution of the RADAR system is in the range of 2 to 3 meters, and the accuracy of location fingerprinting method is superior to signal propagation modeling. In their following work [10], they enhanced RADAR with a Viterbi-like algorithm which improved the accuracy of user location by over 33% and helped alleviate problems due to signal aliasing.

After RADAR, there have been many WLAN location fingerprinting systems developed. Horus [7] system used the probabilistic algorithm in positioning, and the experiment results showed that accuracy reached over 90% probability within 2.1 m. Castro et al. [11] presented a WiFi location service called Nibble, which used Bayesian networks to infer the location of a device. In their experiment setting, the location service reached 97% in accuracy. Taheri et al. [12] described an independent location fingerprinting system, Locus, for comparing the performance of various location fingerprinting algorithms. Kontkanen et al. [13] showed that the probabilistic modeling approach offers a theoretical solution to the positioning problem and many other issues such as calibration, active learning, error estimation, and tracking with history. This solution was also used to develop the Ekahau (http://www.ekahau.com/) commercial WLAN location fingerprinting positioning system. Ching et al. [14] presented the WiPos system using WiFi APs deployed across a university and the system involved developing a server and a client running on the Android platform. Testing showed that university's WiFi network is sufficient to provide "room to room" level accuracy.

In recent years, a number of indoor positioning systems have been developed by integrating other techniques with WLAN location fingerprinting to enhance the indoor positioning performance. Wang et al. [15] presented a floor-map-aided WiFi/pseudo-odometry integration based indoor positioning system and the field experiment showed that it could reliably achieve meter-level accuracy. Karlsson et al. [16] utilized signals of both 2.4 and 5.0 GHz to obtain more information of WiFi, and a particle filter was used to combine location fingerprinting with PDR to improve the accuracy. Ban et al. [17] proposed a high accuracy indoor positioning method by using residual magnetism in addition to PDR and WiFi-based localization methods. They evaluated the method in real environments and confirmed that it could provide accurate indoor positioning with a mean error less than 8 m and more accurate position detection than existing techniques. Kim et al. [18] combined the magnetic field strength, cellular signal strength, and WiFi to build a hybrid system to address some limitations in WiFi localization,

and the experiment demonstrated that the performance was improved in terms of not only accuracy but also computational efficiency.

In addition to the development of different systems, many researchers have conducted research on novel algorithms. Battiti et al. [19] presented a neural network method (the multilayer perceptron) for building a flexible mapping between the raw signal measurements and the position of the mobile terminal. The average accuracy reached approximately 2.3 meters when the environmental changes during the day were taken into account. Youssef et al. [20] provided two implementations for indoor location determination, joint clustering and incremental triangulation, and described the performance regarding accuracy and computation load. The results of their experiment showed that both techniques achieved over 90% accuracy within 7 feet with low computation. Saha et al. [21] assessed the performance of different classifiers including neural network, nearest neighbor, and histogram matching method. The neural network algorithm provided an error of less than one meter with 72% probability, less than 2.6 meters with over 95% probability, and with almost 98% probability it could infer the location within 3.3 meters. A hybrid method that combines the strength of radio frequency propagation loss with location fingerprinting was developed by Kwon et al. in [22]. This hybrid method exhibited 20 to 40% improvement in positioning accuracy to that of competing methods in real site experiment and reached 5 to 7 meters even for a very sparse placement of APs. Ito and Kawaguchi [23] proposed the Bayesian-based location estimation system which achieved an accuracy of 3.0 meters with a cumulative probability of 0.35 in a lecture room and 0.81 in a hallway. Roos et al. [24] studied the WLAN user location estimation problem following a machine learning framework and presented a probabilistic model to solve the estimation problem. In their test, an average location estimation error below 2 meters was easy to obtain, and the two probabilistic methods produced slightly better results than nearest neighbor methods. Yu et al. [25] utilized a support vector machine (SVM) algorithm in the location fingerprinting system and compared it with three kernel functions. Experimental results indicated that the algorithm could improve the localization accuracy and, among the three kernel functions, the radial basis function performs best. Feng et al. [26] presented an accurate RSSI-based indoor positioning system using compressive sensing method, and the experimental results showed that the proposed system leads to substantial improvement in localization accuracy and complexity over the widely used traditional methods. Lu et al. [27] proposed the extreme learning machine with dead zone to address the problems related to signal variations and environmental dynamics in indoor settings. And the real-world experimental results demonstrated that the proposed algorithm could not only provide higher accuracy but also improve the repeatability.

Other research aspects such as fingerprinting database building also attract research interests, especially in recent years. Li et al. [28] presented a method based on Kriging to reduce the workload and save training time and make fingerprinting techniques more flexible and easier to implement. Pan et al. [29] presented a system called LeManCoR based on manifold coregularization, which is a machine learning technique for building a mapping function between data, could adapt the static mapping function effectively, and is robust to the number of RPs. Zhao et al. [30] improved a WiFi fingerprinting based indoor positioning system by efficiently combining the universal Kriging interpolation method, $k$-nearest neighbor ($k$-NN), and naive Bayes classifier, and their lab experiments showed that 28 observation points could achieve the average positioning error of 1.265 m. Ma et al. [31] proposed a fingerprint recovery method based on inexact augmented Lagrange multiplier algorithm. The experiment results indicated that the method could precisely recover the fingerprint and achieve good positioning performance. Liu et al. [32] designed and tested a fast setup algorithm for collecting data for fingerprinting database with the help of smartphone built-in motion sensors. Experiments showed that there was no significant difference on positioning accuracy between the fast setup method and the traditional method. A novel method called RSSI geography weighted regression was proposed by Du et al. [33] to solve the fingerprint database construction problem. The extensive experiments were performed to validate that the proposed method was robust and workforce efficient. Further, two autonomous crowdsourcing systems were proposed to build fingerprint databases on handheld devices by Zhuang et al. [34]. The proposed systems can run on smartphones, build and update databases autonomously, and adaptively account for dynamic environments. Results in different test scenarios indicated that the average positioning errors of both proposed systems are all less than 5.75 m.

Different from the related papers that focus on building individual systems, implementing with different algorithms and constructing fingerprint databases, this paper concerns on impact factors. In the past, some impact factors have been studied. Prasithsangaree et al. [35] presented a study of the positioning performance and placement issues, including algorithm, granularity of the grid in the database, AP fault tolerance, and building architecture. Li et al. [36] discussed pros and cons of different techniques used in WLAN location fingerprinting including numbers of RPs in use, $k$ parameter in the $k$-NN algorithm, distance weighting and universal Kriging in generating fingerprinting database, and probabilistic algorithm in online processing. Honkavirta et al. [37] provided a comparative survey on WLAN RSSI location fingerprinting by introducing the mathematical formulation and tuning the parameters in the formulation. Although papers [35–37] already analyzed some impact factors, to our best of knowledge, there is no systematic summary of the impact factors by using Ishikawa diagrams [9] and experimental analysis of the nonalgorithm factors, which are the main focus of this paper.

## 3. Analysis and Summary of Potential Impact Factors Using the Ishikawa Diagram

This section first introduces the basic quality control tool, Ishikawa diagram, by presenting an example and the diagram

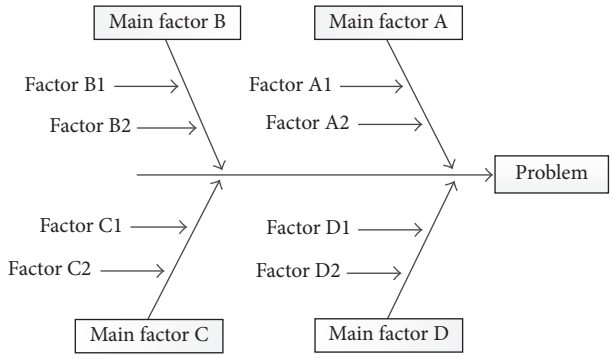



Figure 1: An example of the Ishikawa diagram.

constructing steps, and gives the reasons for choosing this tool. It then constructs an Ishikawa diagram for WLAN location fingerprinting systems and analyzes the potential impact factors to positioning performance in a step-by-step manner, considering WLAN radio wave transmitting, propagating, receiving, and processing.

*3.1. The Ishikawa Diagram Tool.* The Ishikawa diagram [9] was developed by Dr. Kaoru Ishikawa at the University of Tokyo in 1943, who is also a pioneer in quality management techniques in Japan. The diagram has been used in process improvement methods to identify the contributing causes and factors likely to be causing a problem in a systematic way. There are also some other names for Ishikawa diagrams, such as fishbone diagrams, Fishikawa, herringbone diagrams, and cause-and-effect diagrams. Figure 1 shows an example of the Ishikawa diagram, and the name of fishbone diagram comes from its shape. As illustrated in Figure 1, a completed Ishikawa diagram includes a central "spine" and several branches reminiscent of a "fish skeleton." The "fish head" represents the main problem, and the potential factors causing a problem are indicated in the "fish bones" of the diagrams. In constructing an Ishikawa diagram, the first thing is to determine the "fish head" which presents a problem of interest, and the discussion of factors should focus on the problem. The next step is to decide how to categorize the main factors. There are two basic methods to do so, including by function and by process sequence. The third step is to determine the factors in every main factor by analysis, discussion, and summary. Some factors may have subfactors, and all the main factors, factors, and subfactors should be presented in "fish bones."

The Ishikawa diagram is considered one of the seven basic tools of quality control [9]. Moreover, this methodology can be used to any problem and can be tailored by users to fit their circumstances. Other six quality control tools include check sheet, control chart, histogram, Pareto chart, scatter diagram, and stratification, and these six tools emphasize process control or post hoc analysis. However, the Ishikawa diagram is an initial step in the screening process of quality control, which can be a useful technique for organizing some of the information generated in preexperimental planning [38]. After identifying potential causes and factors in a very systematic way, further testing will be necessary to confirm the true causes and factors and their impact pattern. In addition, this method encourages group participation and utilizes group knowledge. The structure provided by the Ishikawa diagram also helps team members to think in a very systematic way and follow a structured approach [39]. Based on the characteristics of the Ishikawa diagram method and the objectives of this paper, the Ishikawa diagram method is adopted.

*3.2. The Ishikawa Diagram Construction for WLAN Fingerprinting.* As previously described, the first thing in constructing an Ishikawa diagram is to present a problem of interest. In this paper, the problem should be the positioning performance of a WLAN RSSI fingerprinting system. The next step is to decide how to categorize the factors. This paper chooses the process sequence in a WLAN RSSI fingerprinting system. Radio signal goes through four steps in location fingerprinting system in either offline or online phase: transmitting, propagating, receiving, and processing. These four steps can be regarded as main factors in the Ishikawa diagram. In the WLAN fingerprinting system, radio signal is first transmitted from APs (online/offline), propagated from transmitters to receivers (online/offline) through environment, received on RPs (offline) or any unknown target location (online), and finally processed to construct fingerprint database (offline) or compute location (online). The positioning performance can be influenced in every step. Therefore, the potential factors will be analyzed step-by-step in detail as follows.

In the signal transmitting step, all factors are about transmitters, which are APs in WLAN RSSI location fingerprinting systems. As previously mentioned, this technique is a cost-saving solution, which means it can make full use of already deployed APs, such as WiFi hotspots. However, hotspots are made by distinct vendors and are diverse with different device parameters such as maximum transmit power. Meanwhile, hotspots may also be placed with different densities, distributions, and heights. As such, a receiver may access different numbers of hotspots in different areas. In summary, differences in APs' device models, density, distribution, and height may influence the RSSI values in the receiver at this step.

At the second step, the radio signal propagates from transmitters to receivers, and RSSI may be influenced by indoor physical environments and the radio signals from other sources. For example, indoor structure, building material, and furniture placement may have impacts on radio signal attenuation, reflection, and diffraction [40]. People crowd in indoor environments may absorb and obstruct the radio signal, and other dynamic objective measures such as temperature and humidity may influence the radio signal propagating as well. Interference may appear between WLAN signals from different APs or other sources such as a microwave oven.

In the signal receiving stage, sampling RSSI in offline learning should be on RPs, whose coordinates should be known previously. The differences of operators, RPs' density, distribution, and height will influence the complexity level and RSSI values in fingerprinting databases. In the online positioning phase, the main work is to search the nearest

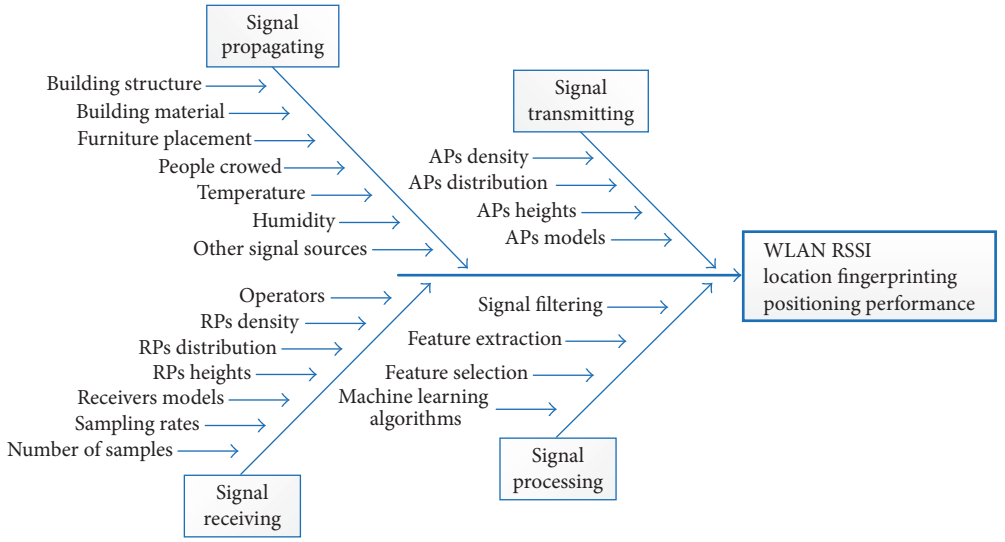



Figure 2: The Ishikawa diagram for impact factors on WLAN RSSI location fingerprinting positioning performance.

RP or RPs by using RSSI sampled on an unknown point, which means that the sampling working is also needed online. Thus, the receivers' model differences in offline and online may have impacts. For example, one smartphone may be utilized in fingerprint database construction usually but users' devices are diverse with different brands and models in online positioning. Additionally, different sampling rate and the number of samples may also influence the samples in fingerprint databases and online positioning results.

The last step is signal processing; the works in this step involve RSSI fingerprint database constructing (offline phase) and location computing (online phase). In database constructing phase, signal filtering method, feature extraction, and selection may change the data and data structure in the fingerprint database. And these factors also determine the features used in online positioning. In online positioning processing, positioning result is computed from received RSSI online and fingerprint database using machine learning algorithms. Hence, the main factors of this step are signal filtering, feature extraction, feature selection, and machine learning algorithm.

To summarize the potential impact factors which influence the WLAN RSSI location fingerprinting performance, an Ishikawa diagram is used in Figure 2. The "fish head" in Figure 2 is WLAN RSSI location fingerprinting positioning performance and "fish bones" are main factors and potential factors.

## 4. Simulation Platform and Factors of Interest Selection

According to Section 3, there are many potential impact factors to WLAN RSSI location fingerprinting positioning performance. To understand the impact pattern between factors and performance, a couple of experiments with different factors settings need to be conducted. For this purpose, this paper builds a simulation platform used as the test environment. The positioning performance should be measurable as well; thus, this paper chooses the root mean square error (RMSE) and cumulative distribution function (CDF) to present the positioning error. For the controllable ability of the simulation platform, simulation feasibility should be taken into consideration when choosing factors of interest for experiments. The potential factors are classified into four types including controllable, uncontrollable, nuisance, and held-constant factors in another Ishikawa diagram, and all the controllable factors are regarded as factors of interest.

*4.1. A WLAN Location Fingerprinting Simulation Platform.* The platform simulates a cuboid test field with 10-meter length, 10-meter width, and 3-meter height. All APs are placed at 3-meter height on the edges of the testing field, and sampling RSSI in offline learning is on grid-shape RPs at one-meter height. Figure 3 shows the placement of four APs in the field and sampling RSSI on a particular RP with coordinates of (5, 5, 1).

Radio signal propagating is described by path loss, which is a positive quantity measured in decibel (dB). The common model for radio signal propagating used in indoor is Log-distance path loss model [41–44] as follows:

$$\mathrm{PL}(d) = \mathrm{PL}(d_0) + 10n \lg\left(\frac{d}{d_0}\right) + X_\sigma, \quad X_\sigma \sim N\left(0, \sigma^2\right), \tag{1}$$

where PL($d$) is the path loss at distance $d$, at near distance $d_0$ is a known received power reference point and $d_0$ is usually chosen as 1 (m) in indoor environment, $n$ is the attenuation factor, and $X_\sigma$ is the signal noise error which obeys a normal distribution with 0 mean value and $\sigma$ standard deviation. Thus, the received power at a distance $d$ can be calculated using

$$P_r = P_t - \mathrm{PL}(d), \tag{2}$$

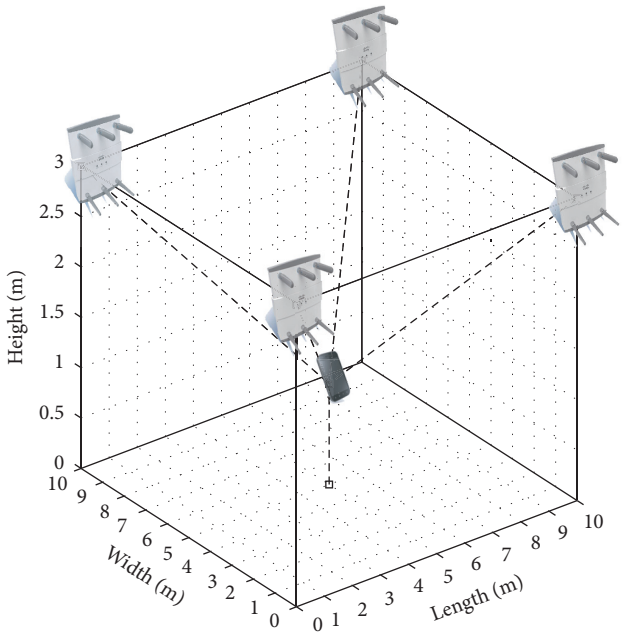



Figure 3: Simulation field with the placement of four APs and one RP.

where $P_t$ is transmitted power and $P_r$ is the received power.

When setting near distance $d_0 = 1$ (m), the path loss can be calculated by free space propagation model in

$$\mathrm{PL}(d) = 10 \lg \frac{P_t}{P_r} = -10 \lg \left[ \frac{G_t G_r \lambda^2}{(4\pi)^2 d^2} \right], \tag{3}$$

where $P_t$ is the transmitted power, $P_r$ is the received power, $G_t$ is the transmitter antenna gain, and $G_r$ is the receiver antenna gain.

For the frequency of WLAN radio signal $f = 2.442$ (GHz), the velocity of radio signal is $c = 3 \times 10^8$ (m/s), and the wavelength is $\lambda = c/f$. Given $G_t = G_r = 1$, we get the $\mathrm{PL}(d_0) = \mathrm{PL}(1) = 40.2$ (dB). Usually, a real AP device's transmitted power $P_t$ is 20 (dBm), which is 100 milliwatts. Thus, the received power $P_r$ measured by RSSI in the simulation work can be calculated by

$$P_r(d) = 20 - \left[40.2 + 10n \lg(d) + X_\sigma\right], \quad X_\sigma \sim N\left(0, \sigma^2\right). \tag{4}$$

As aforementioned, there are two kinds of commonly-used algorithms in fingerprinting location: deterministic and probabilistic methods. Traditional deterministic algorithms can be easily implemented based on $k$-NN. Probabilistic algorithms are based on statistical inference of positioning target and fingerprint database, and they search the positioning result by using the maximum likelihood. Concretely, suppose the number of RPs is $M$, $\mathbf{P}_i$ is the $i$th RP, $\mathbf{S}_i$ is the RSSI fingerprint on $\mathbf{P}_i$, and the number of APs is $N$, $\mathbf{S}_i = (s_{i1}, s_{i2}, \ldots, s_{iN})$, $\mathbf{P}_t$ is positioning target points, and $\mathbf{S}_t$ is the RSSI fingerprint on $\mathbf{P}_t$; thus, $\mathbf{S}_t = (s_{t1}, s_{t2}, \ldots, s_{tN})$. Then, the positioning result $\widehat{\mathbf{P}}_t$ of $\mathbf{P}_t$ can be calculated from formula (5), if it takes the 1-NN as the positioning result. 1-NN is also used by this paper.

$$\widehat{\mathbf{P}}_t = \mathbf{P}_{\arg\min_i \|\mathbf{S}_i - \mathbf{S}_t\|}, \quad i = 1, 2, \ldots, M. \tag{5}$$

In formula (5), the distance is measured by the Euclidean distance of RSSI which is presented as follows:

$$\|\mathbf{S}_i - \mathbf{S}_t\| = \left( \sum_{j=1}^{N} \left| s_{ij} - s_{tj} \right|^q \right)^{1/q}, \quad q = 2. \tag{6}$$

Formula (7) shows the positioning result obtained by the probabilistic algorithm:

$$\widehat{\mathbf{P}}_t = \mathbf{P}_{\arg\max_i P(\mathbf{S}_i \mid \mathbf{S}_t)}, \quad i = 1, 2, \ldots, M. \tag{7}$$

According to Bayes' theorem, the probability can be further transformed into

$$P(\mathbf{S}_i \mid \mathbf{S}_t) = \frac{P(\mathbf{S}_t \mid \mathbf{S}_i) \cdot P(\mathbf{S}_i)}{P(\mathbf{S}_t)} = \frac{P(\mathbf{S}_t \mid \mathbf{S}_i) \cdot P(\mathbf{S}_i)}{\sum P(\mathbf{S}_t \mid \mathbf{S}_i) \cdot P(\mathbf{S}_i)}. \tag{8}$$

In formula (8), $P(\mathbf{S}_t)$ is the same for all RPs searching and $P(\mathbf{S}_i)$ is prior probability and usually regarded as $1/M$. Thus, the positioning result searching is transformed into formula (9).

$$\arg\max_i P(\mathbf{S}_i \mid \mathbf{S}_t) = \arg\max_i P(\mathbf{S}_t \mid \mathbf{S}_i) = \arg\max_i \prod_{j=1}^{N} P\left(s_{tj} \mid \mathbf{S}_i\right). \tag{9}$$

The probability that signal $s_{tj}$ appears on the $i$th RP is $P(s_{tj} \mid \mathbf{S}_i)$, which can be approximately calculated by parametric distributions. In this paper, a Gaussian distribution is selected as shown in formula (10).

$$P\left(s_{tj} \mid \mathbf{S}_i\right) = \frac{1}{\sqrt{2\pi} \cdot \delta} \cdot \exp\left[ -\frac{\left(s_{tj} - \mu\right)^2}{2\delta^2} \right], \tag{10}$$

where $\mu$ and $\delta$ are statistic parameters according to $\mathbf{S}_i$.

*4.2. Output Response and Selection of Factors of Interest.* For the response output of WLAN RSSI location fingerprinting performance, the measurement should be based on positioning error obtained by formula (11):

$$\Delta p = \left\| \mathbf{P}_{\text{true}} - \mathbf{P}_{\text{positioning}} \right\|, \tag{11}$$

where $\Delta p$ is the locating positioning error, $\mathbf{P}_{\text{true}}$ is the true coordinates of a test point, and $\mathbf{P}_{\text{positioning}}$ is the positioning result of the test point.

To facilitate description, this paper chooses the RMSE as the measurement of positioning results, which can be calculated by (12). To give more details about the errors, this

paper chooses the error CDF measured by the cumulative number of test points (CNTP) in the positioning error presentation.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Delta p_i^2}, \tag{12}$$

where $N$ is the total number of test points and $\Delta p_i$ is the $i$th locating positioning error.

According to Montgomery [38], impact factors can be classified as controllable, uncontrollable, nuisance, and held-constant factors, and controllable factors are factors of interest. For the controllable ability of the simulation platform, the simulation feasibility should be taken into consideration when choosing factors of interest for experiments. Additionally, because this paper focuses on nonalgorithm factors, signal filtering, feature extraction, feature selection, and machine learning algorithm are regarded as constant factors. Further, this paper chooses two algorithms including deterministic and probabilistic algorithms during positioning using features of RSSI mean value and standard deviation.

In the signal transmitting step, the density of APs relies on the number in sight. Since the simulation field is a fix $10 \times 10$ ($m^2$) square, the density of APs is measured by the number of APs (APN). The placement heights of APs always relate to building floor height, which means they are similar on the same floor. Therefore, this factor can be regarded as a held-constant factor and be set as 3 (m) in simulation platform. This setting also simplifies APs' distribution problem from three-dimensional to two-dimensional. The two-dimensional APs' distribution can be described by two aspects: (1) distance of APs' centroid to indoor field centroid, which is notated as centroid distance (CD) and (2) the coefficient of variation (CV) of distances from all APs to the centroid, which ban be calculated by formula (8). With regard to AP's models' difference, this paper classifies this factor as nuisance factor which is out the research scope.

$$\text{CV} = \frac{\text{Std}}{\text{Mean}} \times 100\%. \tag{13}$$

Radio signal propagating is described by Log-distance path loss as in formula (1). There are some other propagating models proposed to get more accurate signal strength by improving formula (1) considering floors' and walls' impact to retrieve precise distance for indoor positioning. For example, RADAR [6] system presented a propagating model which takes walls' impact into account. In this paper, the Log-distance path loss model is selected using coefficients $n$ and $\sigma$ to simulate complex indoor environments. It is impracticable to control physical environment impact factors in an experiment platform in reality. However, the influence of indoor structure, building material, and furniture placement can be summarized as signal propagating attenuation factor (notated as $n$) in the radio signal propagating model in the simulation platform. Meanwhile, dynamic people crowd, temperature, humidity, and other signal resources can be regarded as the signal noise standard deviation (notated as $\sigma$) in the propagating model as well. Thus, attenuation factor and signal noise can be chosen as controllable factors in the signal propagating step, and the empirical values of $n$ and $\sigma$ are presented in Table 1 according to Rappaport [41].

Table 1: Empirical coefficient values for indoor propagation in various types of buildings.

| Building type | $n$ | $\sigma$ |
|---|---|---|
| Vacuum, infinite space | 2.0 | 0 |
| Retail store | 2.2 | 8.7 |
| Grocery store | 1.8 | 5.2 |
| At same floor | 2.8 | 12.9 |
| Through three floors | 5.2 | 6.7 |
| Office | 2.7 | 8.1 |
| Office | 3.2 | 11.2 |
| Office | 3.2 | 4.4 |
| Office | 3.5 | 12.8 |
| Office | 4.0 | 4.3 |
| Factory | 2.1 | 9.7 |
| Factory | 3.3 | 6.8 |

For the signal receiving, the fingerprint sampling can be operated on RPs in the grid shape. Thus, the distribution can be regarded as held-constant factors, and the density of RPs can be measured by RPs grid interval distance (GID). In the simulation platform, RPs' heights, sample rate, and the number of samples are also regarded as held-constant factors and are set as 1 (m), 1 (Hz), and 240 (four orientations), respectively. Nevertheless, operators may be a person or a robot with different receiver models, and these two factors are complex in reality. In this paper, they are classified as nuisance factors.

Figure 4 illustrates the response output and classified factors using another Ishikawa diagram. It should be noted that a controllable factor remains constant and does not change throughout an experiment test, while the uncontrollable one means a factor varies randomly during the test. In reality, $\sigma$ factor in radio signal propagating is an uncontrollable factor, but for the controllability of the simulation platform, $\sigma$ is regarded as a controllable factor as shown in Figure 4. Table 2 lists all the controllable factors (factors of interest), measurement details, and notations.

## 5. Experiment and Results

According to Section 4, a simulation platform was built for experiment and it took five controllable factors as factors of interest. In the experiment, the OFAT (one-factor-at-a-time) analysis method was adopted, which omits the interaction impact of all controllable factors. This section describes the details of experiment settings and results.

*5.1. Tests Settings in the Analysis Experiment.* To conduct the OFAT analysis experiment, a couple of tests with different settings of controllable factors were conducted in the simulation system. Table 3 shows all the experiment settings. It should be noted that a (start, end, and step) notation was used to describe a level range and a {levels} notation was used to

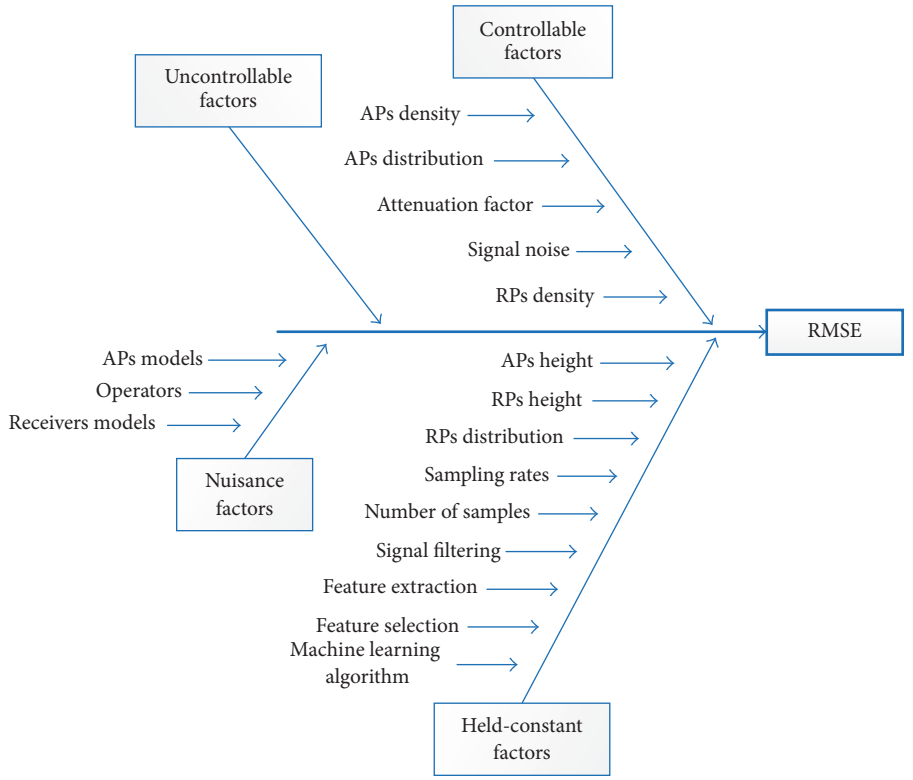



Figure 4: The Ishikawa diagram for factor classification and response output.

Table 2: All the controllable factors (factors of interest) and their measurement details.

| Controllable factor | Measurement detail | Measurement notation |
|---|---|---|
| APs density | Number of APs in the test area | APN |
| APs distribution | Distance of APs' centroid to indoor field centroid, and the coefficient of variation of distances from all APs to the centroid | CD and CV |
| Attenuation factor | Attenuation factor in simulation radio signal propagating model | $n$ |
| Signal noise | Signal noise standard deviation in simulation radio signal propagating model | $\sigma$ |
| RPs density | Grid interval distance of RPs when sampling RSSI fingerprint | GID |

Table 3: Settings for all the factors of interest (controllable factors) in the OFAT analysis experiment.

| Test ID | Factors | APs density (APN) | APs distribution (CD (m)/CV) | Attenuation factor ($n$) | Signal noise ($\sigma$ (dB)) | RPs density GID (m) |
|---|---|---|---|---|---|---|
| 1 | APs density (APN) | *Test factor (1, 20, 1)* | Open-shape | 2 | {1, 5, 8, 10, 15} | 1 |
| 2 | APs distribution (CD (m)/CV) | 5 | *Test factor (110 different shapes)* | 2 | {1, 5, 8, 10, 15} | 1 |
| 3 | Attenuation factor ($n$) | {3, 5, 10, 15, 20} | Open-shape | *Test factor (1, 5.5, 0.5)* | 5 | 1 |
| 4 | Signal noise ($\sigma$ (dB)) | {3, 5, 10, 15, 20} | Open-shape | 2 | *Test factor (1, 15, 1)* | 1 |
| 5 | RPs density GID (m) | 5 | Open-shape | 2 | {1, 5, 8, 10, 15} | *Test factor {0.1, 0.2, 0.5, 1, 1.25, 2, 2.5, 5}* |

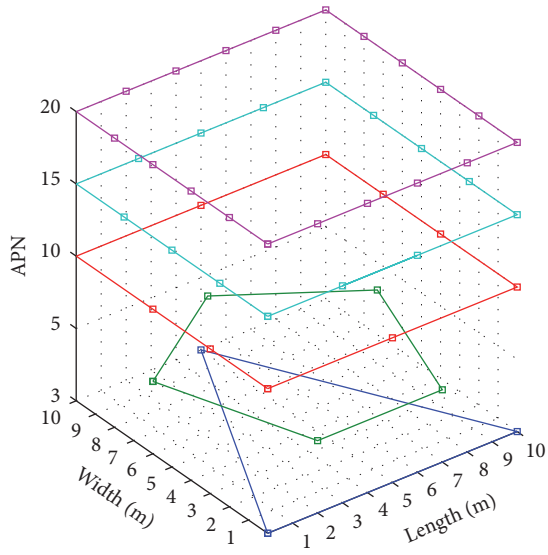



Figure 5: The open-shape distributions for five different APs densities.

describe different factor levels. For example, in the first test, APN was set with 20 levels as $\{1, 2, \ldots, 20\}$ and $\sigma$ was set with five levels as $\{1, 5, 8, 10, 15\}$. Meanwhile, it is inconvenient to show the CD and CV of APs distribution in a single table cell because CD and CV depend on coordinates of APs, which varies in every test. Therefore, this paper uses the "open-shape" to describe APs distribution in the experiment when the distribution is not the test factor. The "open-shape" APs distribution prefers a dispersed shape, which simulates the placement of APs in a real environment for communication. Figure 5 shows the open-shape APs placement when APN is set as 3, 5, 10, 15, and 20, which are common settings in Table 3. It also should be noted that empirical coefficient values for indoor propagation in Table 1 were taken into consideration in all settings for attenuation factor and signal noise.

According to Table 3, the first test factor was APs density with APN measurements, and in the APs density test, APN changed from 1 to 20 with a step of 1 AP and every APN setting took the open-shape APs distribution. The attenuation factor $n$ was set as 2, signal noise $\sigma$ changed in five levels that is, 1, 5, 8, 10, and 15 (dB), and GID was 1 (m).

The second test factor was APs distribution measured by CD and CV, and it took 110 different shapes of five APs into consideration. In APs distribution test, APN was 5, $n$ was 2, $\sigma$ ranged in five levels 1, 5, 8, 10, and 15 (dB), and GID was 1 (m). Figure 6 demonstrates all the 110 different shapes in the simulation fields.

With consideration of empirical coefficient values for indoor propagation in Table 1, the next test factor was attenuation factor $n$, and $n$ ranged from 1 to 5.5 with an interval of 0.5 in the test. The APs were placed in the open-shape and the values changed into 3, 5, 10, 15, and 20, the $\sigma$ was set as 5 (dB), and the GID was set as 1 (m).

The test factor, signal noise standard deviation $\sigma$, ranged from 1 to 15 with a step of 1 (dB) in the signal noise test. The APs were deployed in the same way as with the attenuation factor test, and $n$ was 2 and GID was 1 (m).

The last controllable test factor was RPs density described by GID between two neighboring RPs. In the test of this factor, the GID varies with eight different settings, which were 0.1, 0.2, 0.5, 1, 1.25, 2, 2.5, and 5 (m). Figure 7 shows the different RPs distributions with different GID values. Other factors were set as follows: five APs in open-shape, $n$ was 2, and $\sigma$ had five levels including 1, 5, 8, 10, and 15 (dB).

In every OFAT test, there was one fact under test and the procedure on the simulation platform is as follows: (1) input coordinates of APs, GID, and test points number (set as 100 in this study), (2) generate RPs and the fingerprint database of RPs, (3) generate the test points randomly and generate the signal strength of all the test points, (4) test all the testing points with deterministic algorithm, (5) test all the testing points with probabilistic algorithm, and (6) output the error, calculate the statistics, and plot the RMSE and CDF graphs. These steps are shown in Figure 8.

### 5.2. Tests Results in the Analysis Experiment

*5.2.1. APs Density.* Figure 9 shows the test results of positioning error by applying the deterministic and probabilistic algorithms on different APs densities. Figure 9(a) shows the trend of RMSE on different APNs in the simulation test area with different $\sigma$ levels when $n = 2$ and GID = 1 (m). In this figure, we can find that the more APs it can access, the better positioning result it can achieve when the noise error is low. But if the noise error is high, for example, $\sigma = 15$ (dB), APN becomes less important. It should be noted that $\sigma = 1$ (dB) approximates an ideal noise situation, and in this ideal situation, RMSE is high when APN is less than 3 and the RMSE declines gently when APN increases from 3 to 20. This line trend indicates that at least 3 APs are needed for RSSI location fingerprinting theoretically. Figure 9(b) gives more details about the error distribution with different $\sigma$ levels, and the line located in "up-left" of CDF means better accuracy performance. It can be seen from Figure 9(b) that lines with higher APN are more towards "up-left" than ones with lower APN, but with the increase of noise error, the error lines become more towards "down-right" and close to each other. These lines indicate that large APN can reduce the positioning error, but if the noise becomes higher, the influence of APN becomes less and less.

*5.2.2. APs Distribution.* Figure 10 shows the test results of positioning errors by using the deterministic and probabilistic algorithms on 110 different APs distributions measured by CD and CV with model setting as APN = 5, $n = 2$, and GID = 1 (m). Figures 10(a) and 10(c) plot the RMSE on CD and CV, respectively, at different $\sigma$ levels; however, we can hardly find clear patterns on RMSE with CD and CV in these two figures except that higher $\sigma$ have higher RMSE levels. Figures 10(b) and 10(d) plot the CNTP with ordered CD and CV. In these two figures, we can see that lines with lower $\sigma$ are more towards "up-left" than the higher $\sigma$, but no matter $\sigma$ is, all the lines are unordered with CD and CV. Thus, similar to RMSE,

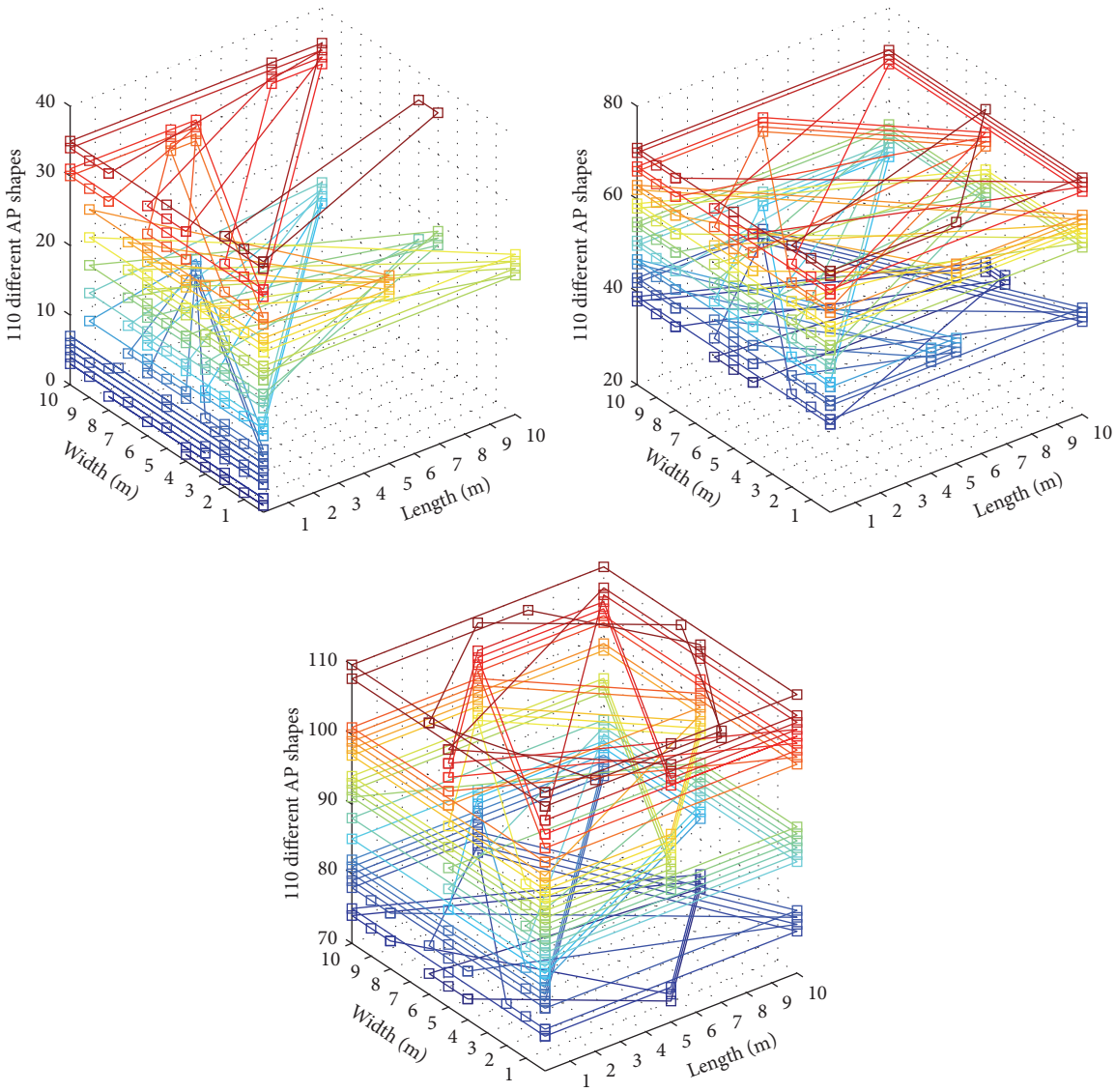



Figure 6: The 110 different shapes of five APs placement.

we can hardly summarize the influence trend of CD and CV to error distribution according to Figures 10(b) and 10(d).

*5.2.3. Attenuation Factor.* Figure 11 illustrates the test results of positioning errors by using the deterministic and probabilistic algorithms on attenuation factor $n$. Figure 11(a) shows the RMSE versus $n$ under $\sigma = 5$ (dB) and GID = 1 (m) at different APN levels. According to this figure, we can see that the RMSE declines with the increase of $n$. And the larger APN shows lower RMSE level: for example, line with APN = 20 locate lower than APN = 3. But APN's impact trends to less when APN has been already large; for instance, lines are closer to each other when APN is set to 10, 15, and 20 comparing to the settings of 3 and 5. Error distributions are shown in Figure 11(b); it can be seen that lines with higher $n$ and more APs are more towards "up-left," which means higher n and more APs are good for better performance. However, error distributions are similar when APN is large, such as 10, 15, and 20, no matter which algorithm is used.

*5.2.4. Signal Noise.* Test results of positioning errors by using the deterministic and probabilistic algorithms on signal noise $\sigma$ with model settings $n = 2$ and GID = 1 (m) are illustrated in Figure 12. The RMSE results with $\sigma$ under different APN levels are shown in Figure 12(a). There are growing trends of RMSE results with the $\sigma$ level in all APN settings in Figure 12(a), which means that the higher noise level in the environment, the worse positioning performance. It is interesting to see that as $\sigma$ increases, all the lines change from close through apart to close again. These lines indicate that APN can hardly influence the result when the environment has very low or high lever noise, for example, when $\sigma$ is lower than 3 (dB) or higher than 13 (dB) in Figure 12(a). Figure 12(b) shows the error distributions of positioning results. In this figure, signal

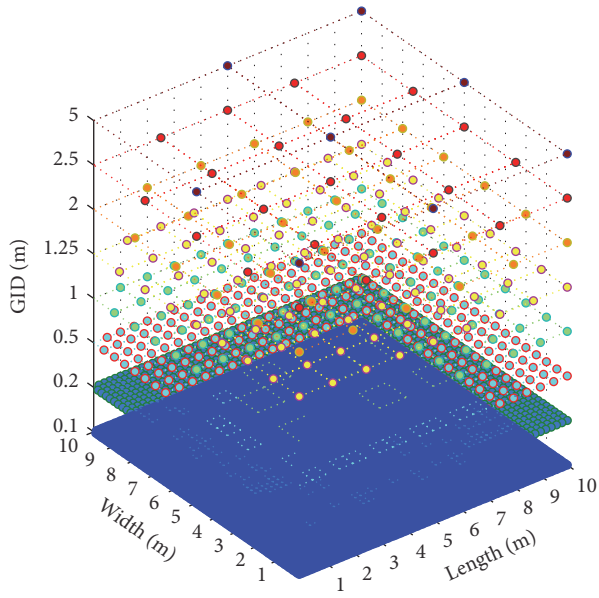



Figure 7: Different RPs distributions with different GID values.

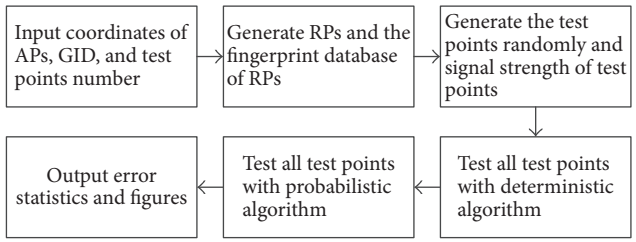



Figure 8: The flowchart of a factor test using the simulation platform.

noise influence can be clearly seen from the color lines in every error distribution; however, the impact of APN is not clear for the similar error distributions especially when APN is larger than 3.

*5.2.5. RPs Density.* Test results of positioning errors by using the deterministic and probabilistic algorithms on RPs density GID are shown in Figure 13. The model was set as APN = 5 and $n = 2$, and Figure 12(a) illustrates the RMSE results on GID with different $\sigma$ levels. All the lines of RMSE show increasing trends, but this trend becomes smoother when $\sigma$ is high. This phenomenon is much clearer when taking the error distributions in Figure 12(b) into consideration. When $\sigma = 1$ (dB), all the lines of CNTP with less GID are located "up-left" no matter which algorithm is used. However, if $\sigma$ increases, these lines tend to move "down-right" and much nearer. According to Figure 13, reducing GID will not benefit positioning performance significantly when GID is less than 1 (m) and especially when noise is low as $\sigma = 1$ (dB). It seems that 1.25 (m) is a good choice for GID, if noise is high; for example, $\sigma = 15$ (dB).

## 6. Conclusions and Future Work

WLAN location fingerprinting is a cost-saving solution but it still faces difficulties in practical applications because of the impact of the factors such as inhomogeneous APs placement, unstationary WLAN RSSI, and additional offline learning work. There are many factors that influence positioning performance in location fingerprinting systems. A good summary of potential factors and analysis of their impact patterns can benefit the applications and quality control of location fingerprinting. The issue is challenging; however, little effort has been made on systematical investigation.

This paper analyzed the impact factors of positioning performance in RSSI location fingerprinting systems step-by-step considering the radio transmitting, propagating, receiving, and processing and summarized potential factors by using Ishikawa diagram to provide a reference for further research. To facilitate the analysis experiment of factors and impacts, this paper presented a simulation WLAN location fingerprinting platform. The paper classified all the factors into controllable, uncontrollable, nuisance, and held-constant factors in another Ishikawa diagram with consideration of the feasibility of the simulation platform. Finally, the paper considered five controllable factors (including APs density, APs distribution, radio signal propagating attenuation factor, radio signal propagating noise, and RPs density) as factors of interest and utilized the OFAT analysis method to conduct the experiment to reduce the complexity and number of tests in a factor analysis experiment.

The results indicate that high APs density, signal propagating attenuation factor, and RPs density with a low level of signal propagating noise are favorable for better positioning performance, while APs distribution has no particular impact pattern on the performance. It is not necessary to improve RPs density to get better positioning performance when GID is less than 1 meter. Moreover, high RPs density means a heavy work for building fingerprinting database.

According to the results, some observations can be drawn to guide the quality control in applications of WLAN location fingerprinting: (1) the number of APs must be larger than 3 for location fingerprinting, and deploying some external APs can improve the performance especially in a noisy environment no matter what the distribution these APs have. However, if the environment is very noisy (e.g., $\sigma$ is about 15 (dB) according to the experiments), this method can hardly be effective; (2) an environment with complex structures, such as a unit with many walls and rooms (high attenuation factor), is more preferable for deploying location fingerprinting than a simple one like an underground parking lot (low attenuation factor); and (3) improving RPs density can benefit positioning, but it is useless when the GID is less than 1 meter. Meanwhile, high RPs density means more work for fingerprinting database building and updating.

In the near future, real field experiments will be designed and conducted to further verify the conclusions given in this paper. It should be noted that to reduce the complexity and number of tests in a factor analysis experiment, an OFAT method was used to omit the factor interactions. Although

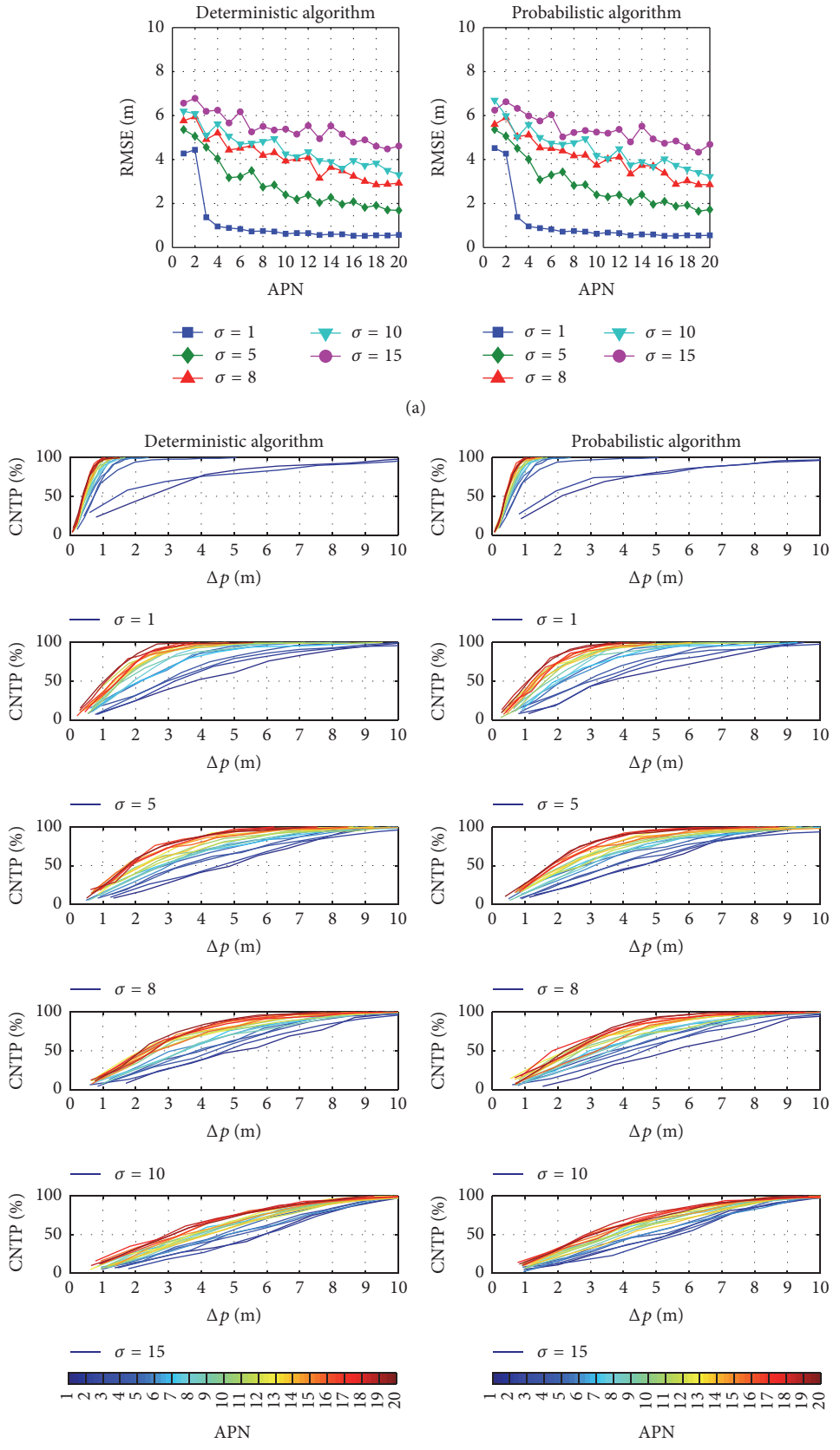
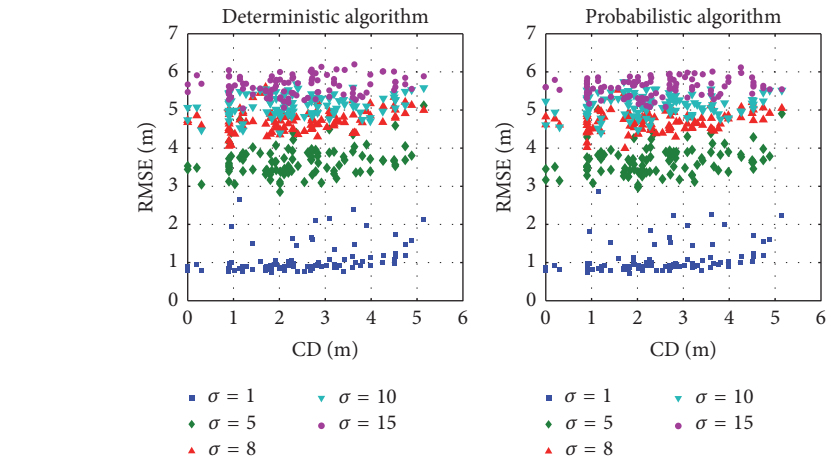
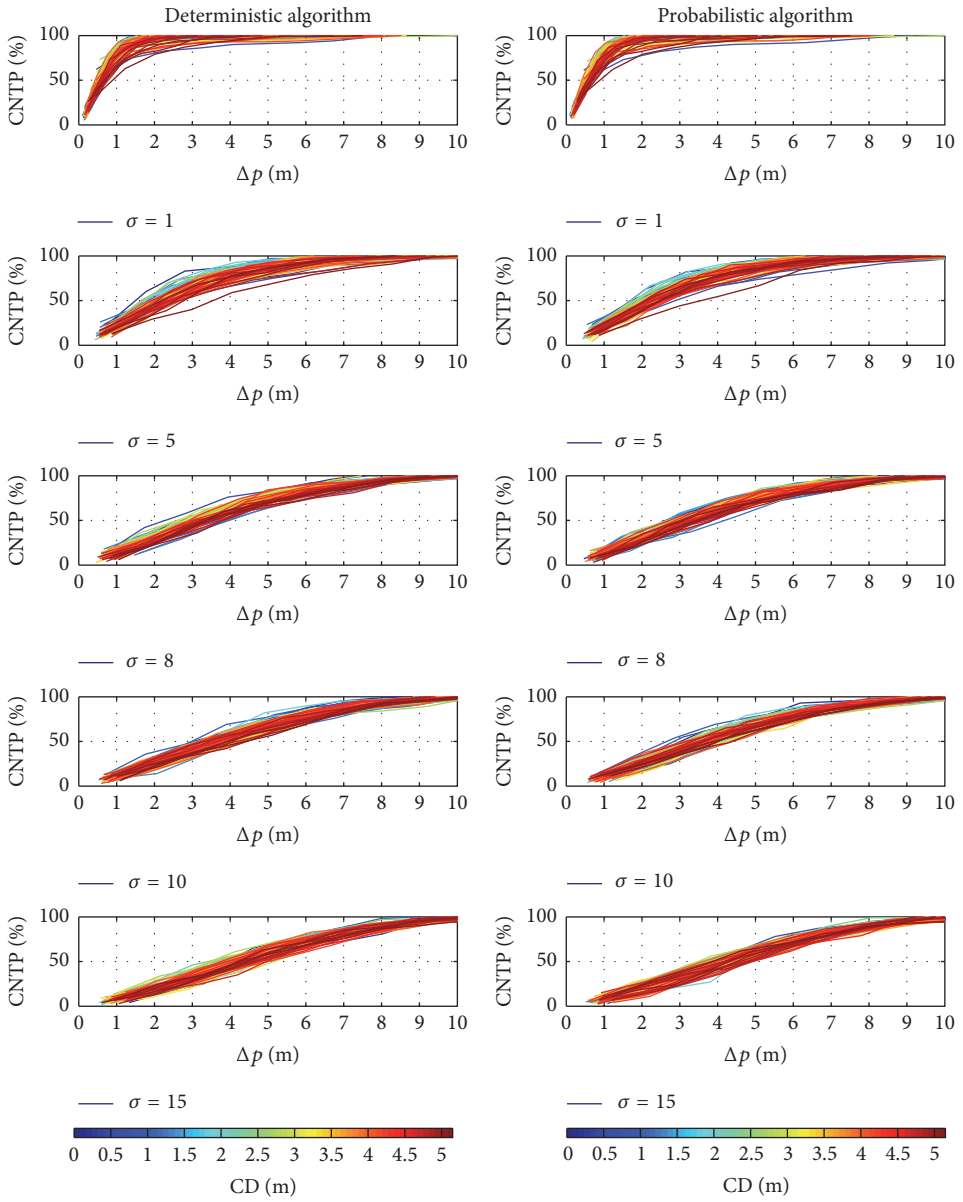



Figure 9: Test results of positioning error by using the deterministic and probabilistic algorithms on different APs densities including (a) RMSE versus APN and (b) CNTP versus APN.




(a)




(b)

Figure 10: Continued.

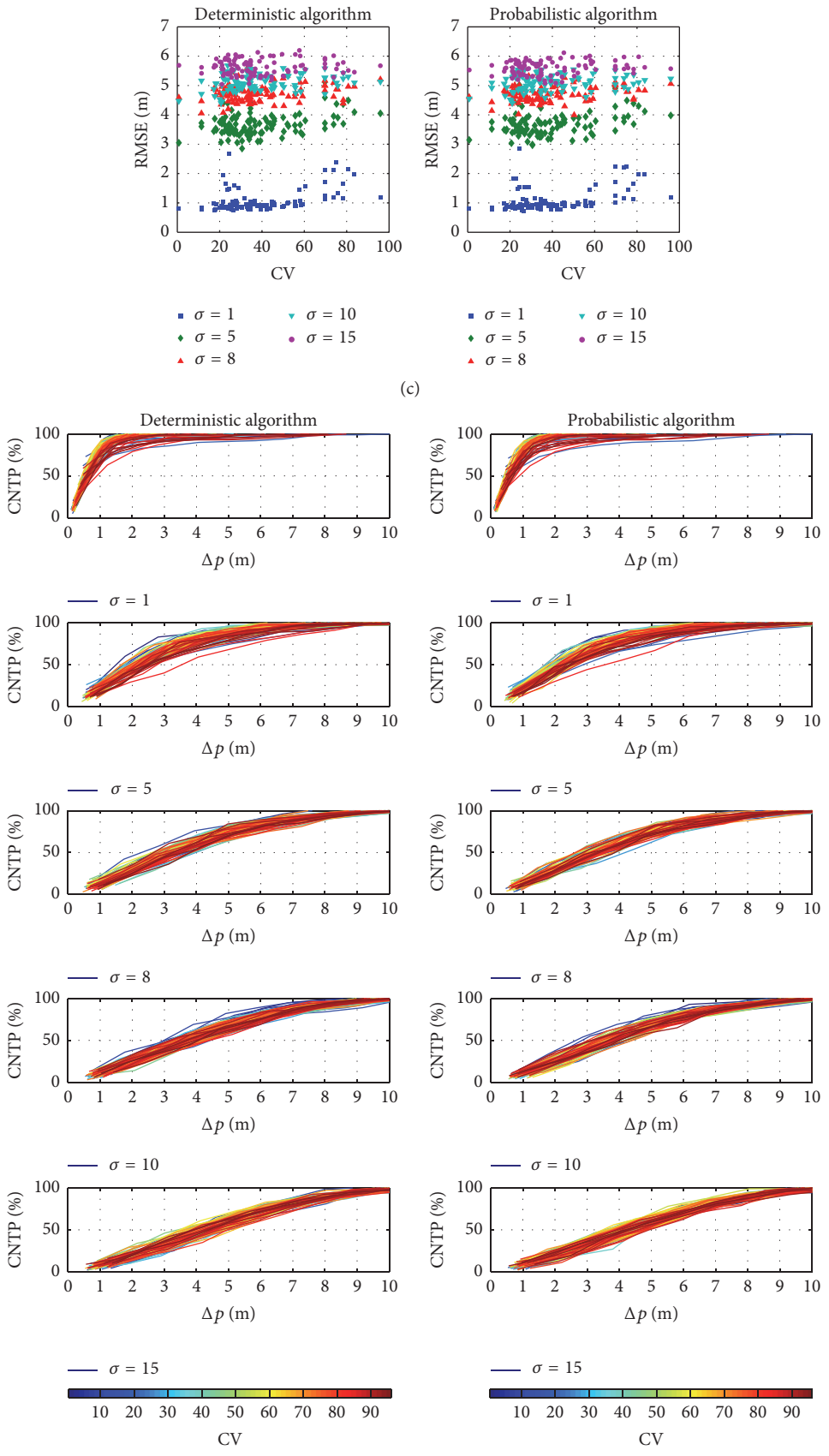
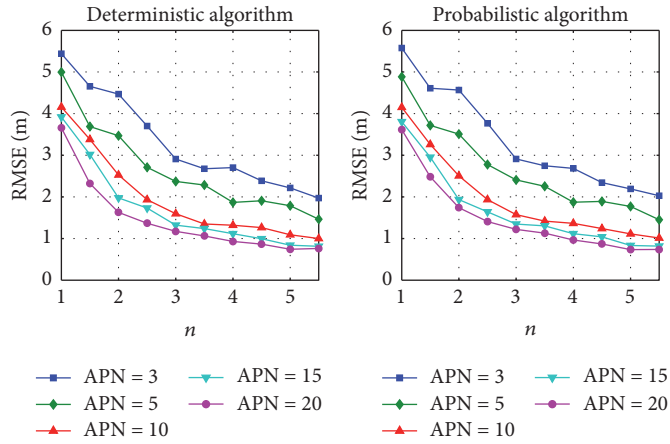
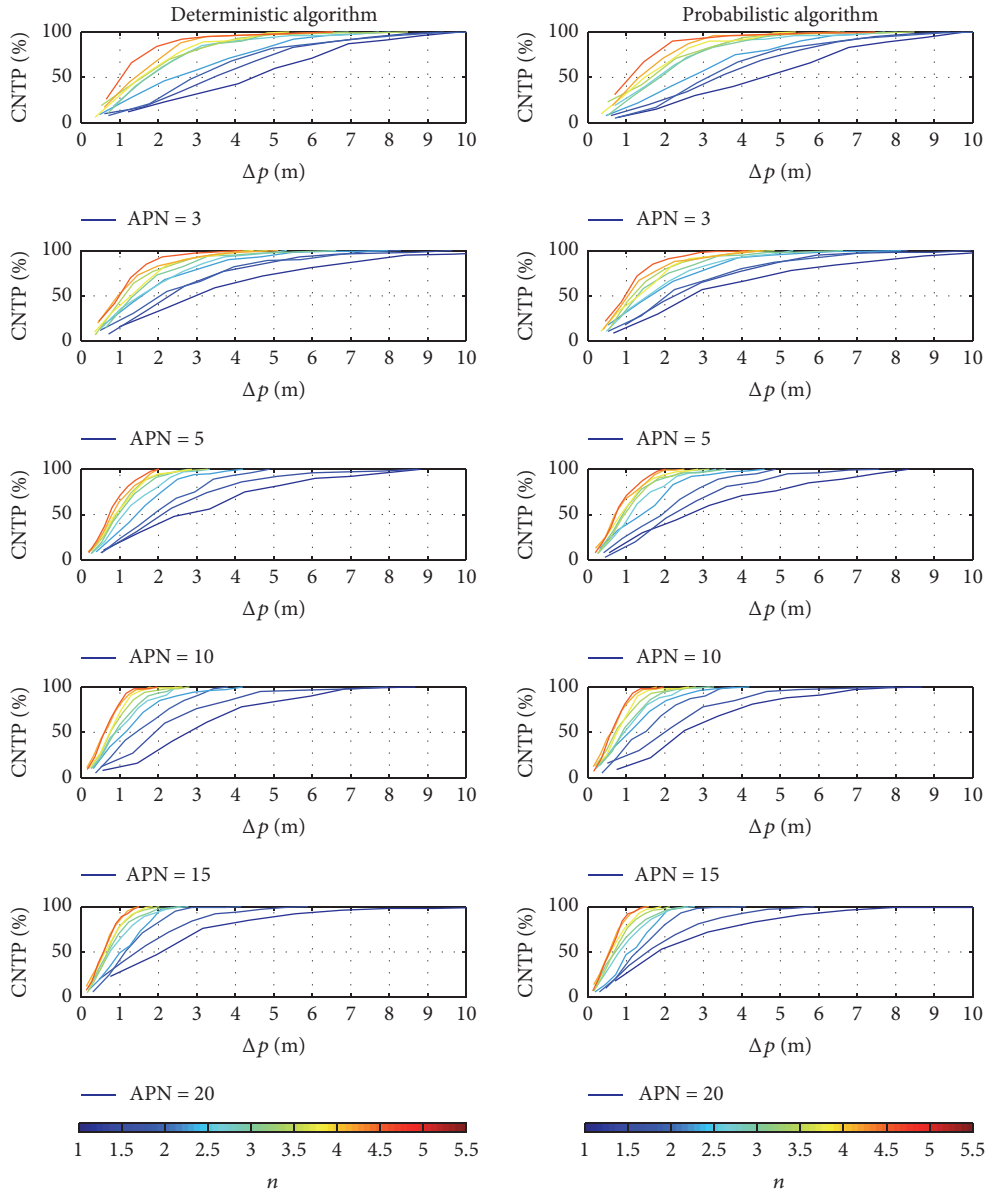



Figure 10: Test results of positioning errors by using the deterministic and probabilistic algorithms on different APs distributions including (a) RMSE versus CD, (b) CNTP versus CD, (c) RMSE versus CV, and (d) CNTP versus CV.




(a)




(b)

Figure 11: Test results of positioning errors by using the deterministic and probabilistic algorithms on different attenuation factors including (a) RMSE versus $n$ and (b) CNTP versus $n$.

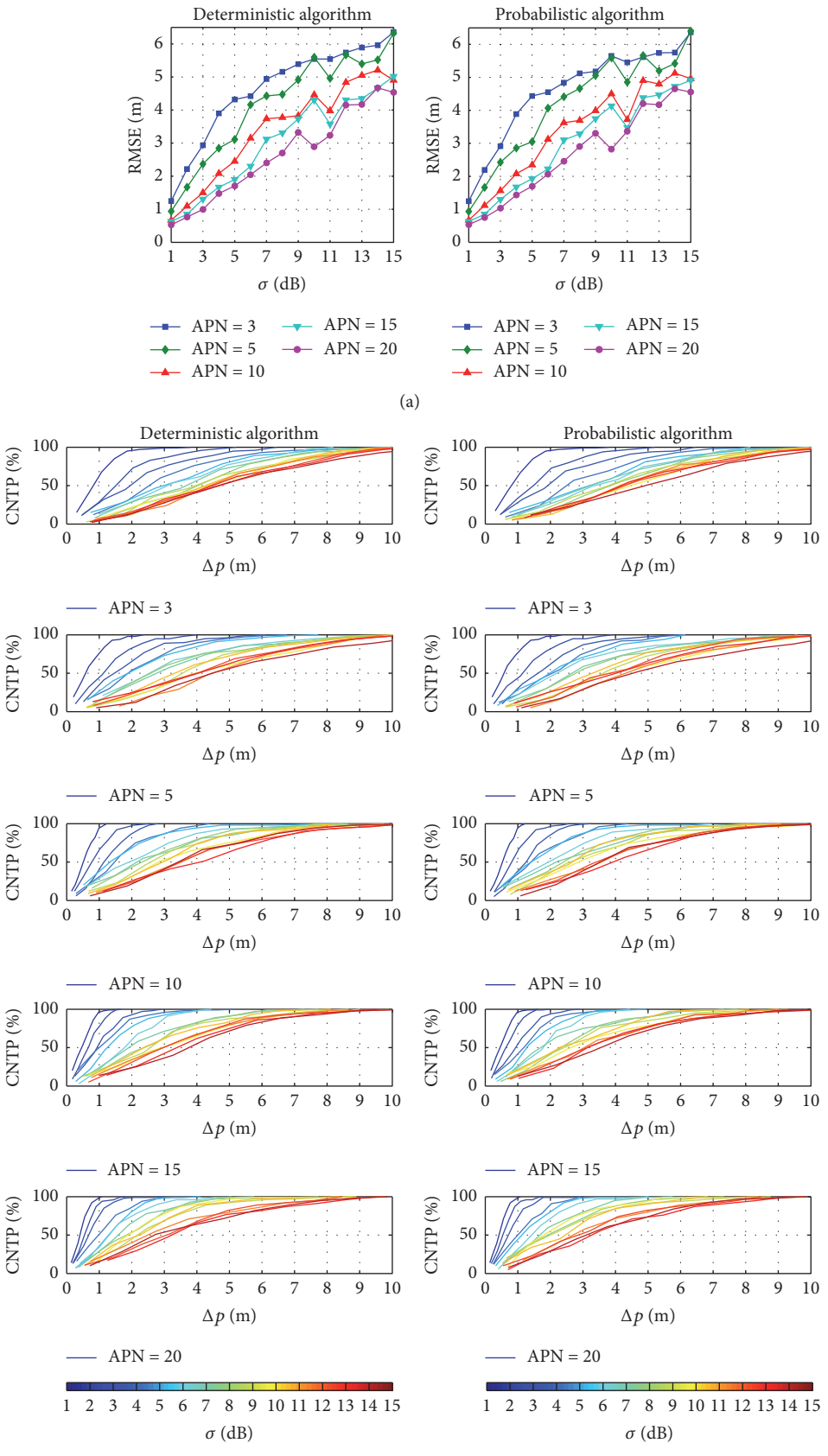



FIGURE 12: Test results of positioning errors by using the deterministic and probabilistic algorithms on different signal noise errors including (a) RMSE versus $\sigma$ and (b) CNTP versus $\sigma$.

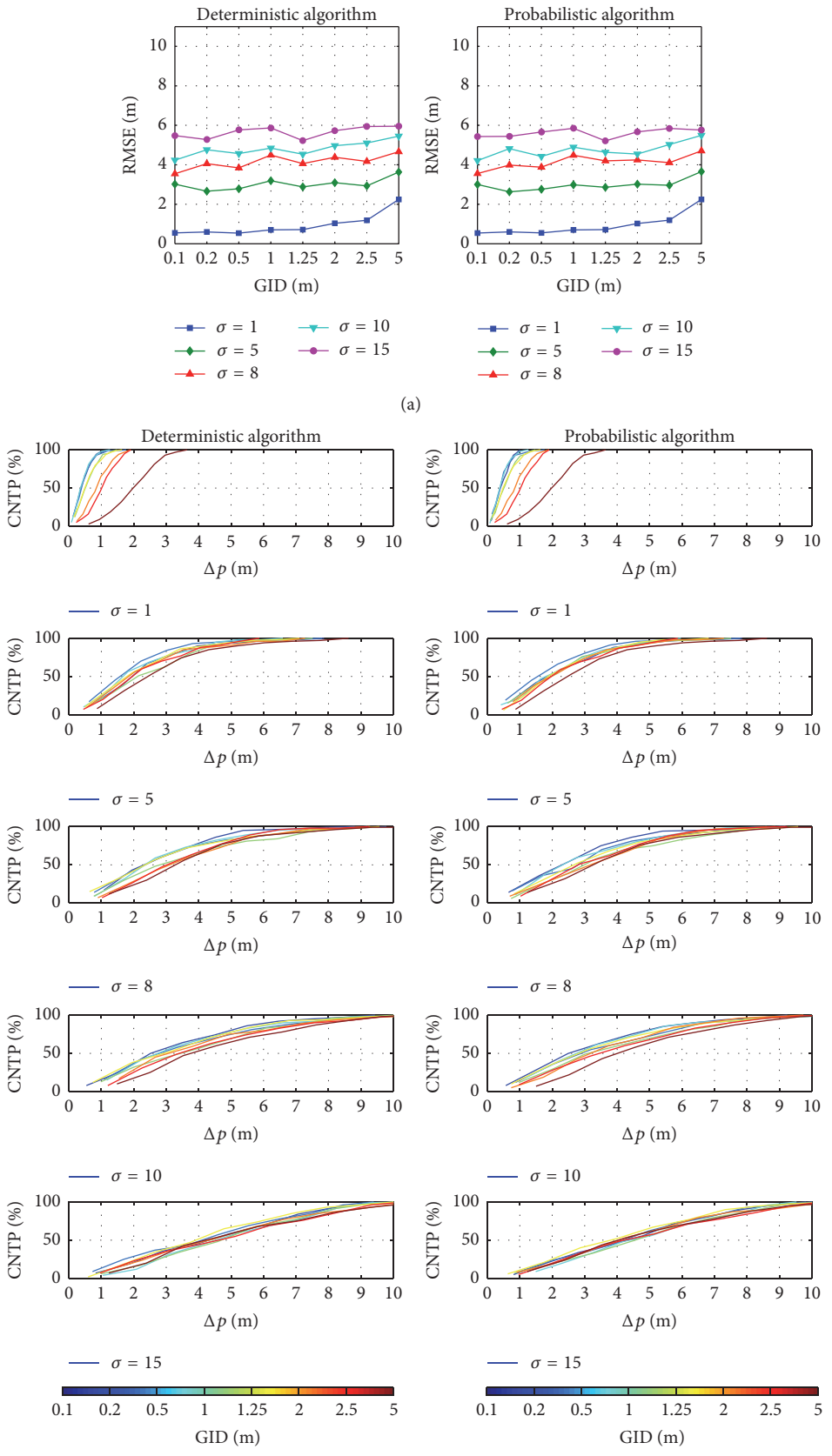



FIGURE 13: Test results of positioning errors by using the deterministic and probabilistic algorithms on different RPs densities including (a) RMSE versus GID and (b) CNTP versus GID.

it can demonstrate two factors' interactions in every test, a more complicated factor analysis experiment needs to be conducted to extend the OFAT experiment in the future. We also plan to conduct practical applications of WLAN location fingerprinting and control the quality of positioning results based on the outcomes of this paper.

## Acronyms

WLAN: Wireless local area network
AP: Access point
OFAT: One-factor-at-a-time
GPS: Global positioning system
*k*-NN: *k*-nearest neighbor
CNTP: Cumulative number of test points
CD: Centroid distance
GID: Grid interval distance
RSSI: Received signal strength indicator
RP: Reference point
RMSE: Root mean square error
PDR: Pedestrian dead reckoning
CDF: Cumulative distribution function
APN: APs number
CV: Coefficient of variation.

## Competing Interests

The authors declare no conflict of interests.

## Authors' Contributions

Keqiang Liu and Yunjia Wang conceived the paper and designed the experiments. Keqiang Liu summarized the impact factors, developed simulation platform, and conducted the analysis experiment. Lixin Lin and Guoliang Chen participated in analyzing the data. Keqiang Liu mainly composed this paper, and Yunjia Wang, Lixin Lin, and Guoliang Chen contributed in revising. All authors participated in elaborating the paper and proofreading.

## Acknowledgments

The research work presented in this paper is partly supported by China National Key Research and Development Program (Grant no. 2016YFB0502102), the Natural Science Foundation of Jiangsu Province (no. BK20161181), the Key Laboratory of Advanced Engineering Surveying of National Administration of Surveying, Mapping and Geo-Information (Grant no. TJES1302), Education Department of Jiangsu (Grant no. KYLX_1394), the National Natural Science Foundation of China (no. 41371423), and the Priority Academic Program Development of Jiangsu Higher Education Institutions (Grant no. SZBF2011-6-B35). All the authors would like to thank Professor Songnian Li from Ryerson University, Canada, for helping them in English writing which greatly improved the manuscript.

## References

[1] P. A. Zandbergen, "Accuracy of iPhone locations: a comparison of assisted GPS, WiFi and cellular positioning," *Transactions in GIS*, vol. 13, no. 1, pp. 5–25, 2009.

[2] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, 2007.

[3] L. Pei, R. Chen, J. Liu, H. Kuusniemi, T. Tenhunen, and Y. Chen, "Using inquiry-based Bluetooth RSSI probability distributions for indoor positioning," *Journal of Global Positioning Systems*, vol. 9, pp. 122–130, 2010.

[4] L. Chen, P. Thevenon, G. Seco-Granados, O. Julien, and H. Kuusniemi, "Analysis on the TOA tracking with DVB-T signals for positioning," *IEEE Transactions on Broadcasting*, vol. 62, no. 4, pp. 957–961, 2016.

[5] J. Liu, R. Chen, L. Pei, R. Guinness, and H. Kuusniemi, "A hybrid smartphone indoor positioning solution for mobile LBS," *Sensors (Switzerland)*, vol. 12, no. 12, pp. 17208–17233, 2012.

[6] P. Bahl and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '00)*, pp. 775–784, Tel Aviv, Israel, March 2000.

[7] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys '05)*, pp. 205–218, ACM, Seattle, Wash, USA, June 2005.

[8] L. Chen, L. Pei, H. Kuusniemi, Y. Chen, T. Kröger, and R. Chen, "Bayesian fusion for indoor positioning using bluetooth fingerprints," *Wireless Personal Communications*, vol. 70, no. 4, pp. 1735–1745, 2013.

[9] K. Ishikawa, *Guide to Quality Control*, Asian Productivity Organization, Tokyo, Japan, 1976.

[10] P. Bahl and V. N. Padmanabhan, "Enhancements to the RADAR user location and tracking system," Tech. Rep. MSR-TR-2000-12, Microsoft Research, Redmond, Wash, USA, 2000.

[11] P. Castro, P. Chiu, T. Kremenek, and R. Muntz, "A probabilistic room location service for wireless networked environments," in *Proceedings of the 3rd International Conference on Ubiquitous Computing (UbiComp '01)*, G. D. Abowd, B. Brumitt, and S. A. Shafer, Eds., pp. 18–34, Springer, Atlanta, Ga, USA, September-October 2001.

[12] A. Taheri, A. Singh, and E. Agu, "Location fingerprinting on infrastructure 802.11 Wireless Local Area Networks (WLANs) using locus," in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN '04)*, Tampa, Fla, USA, November 2004.

[13] P. Kontkanen, P. Myllymäki, T. Roos, H. Tirri, K. Valtonen, and H. Wettig, "Topics in probabilistic location estimation in wireless networks," in *Proceedings of the IEEE 15th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '04)*, pp. 1052–1056, IEEE, Barcelona, Spain, September 2004.

[14] W. Ching, R. J. Teh, B. Li, and C. Rizos, "Uniwide WiFi based positioning system," in *Proceedings of the IEEE Internationl Symposium on Technology and Society: Social Implications of Emerging Technologies (ISTAS '10)*, pp. 180–189, Wollongong, Australia, June 2010.

[15] J. Wang, A. Hu, C. Liu, and X. Li, "A floor-map-aided WiFi/pseudo-odometry integration algorithm for an indoor positioning system," *Sensors (Switzerland)*, vol. 15, no. 4, pp. 7096–7124, 2015.

[16] F. Karlsson, M. Karlsson, B. Bernhardsson, F. Tufvesson, and M. Persson, "Sensor fused indoor positioning using dual band WiFi signal measurements," in *Proceedings of the European Control Conference (ECC '15)*, July 2015.

[17] R. Ban, K. Kaji, K. Hiroi, and N. Kawaguchi, "Indoor positioning method integrating pedestrian dead reckoning with magnetic field and wifi fingerprints," in *Proceedings of the 8th International Conference on Mobile Computing and Ubiquitous Networking (ICMU '15)*, pp. 167–172, IEEE, Hakodate, Japan, January 2015.

[18] B. Kim, M. Kwak, J. Lee, and T. T. Kwon, "A multi-pronged approach for indoor positioning with WiFi, magnetic and cellular signals," in *Proceedings of the 5th International Conference on Indoor Positioning and Indoor Navigation (IPIN '14)*, pp. 723–726, Busan, Korea, October 2014.

[19] R. Battiti, T. Nhat, and A. Villani, "Location-aware computing: a neural network model for determining location in wireless LANs," Tech. Rep. DIT-02-083, University of Trento, Trento, Italy, 2002.

[20] M. A. Youssef, A. Agrawala, and A. U. Shankar, "WLAN location determination via clustering and probability distributions," in *Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications (PerCom '03)*, pp. 143–150, Fort Worth, Tex, USA, March 2003.

[21] S. Saha, K. Chaudhuri, D. Sanghi, and P. Bhagwat, "Location determination of a mobile device using IEEE 802.11b access point signals," in *Proceedings of the IEEE Wireless Communications and Networking Conference: The Dawn of Pervasive Communication (WCNC '03)*, vol. 3, March 2003.

[22] J. Kwon, B. Dundar, and P. Varaiya, "Hybrid algorithm for indoor positioning using wireless LAN," in *Proceedings of the IEEE 60th Vehicular Technology Conference (VTC' 04)*, pp. 4625–4629, Los Angeles, Calif, USA, September 2004.

[23] S. Ito and N. Kawaguchi, "Bayesian based location estimation system using wireless LAN," in *Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom 2005 Workshops*, pp. 273–278, Kauai Island, Hawaii, USA, 2005.

[24] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen, "A probabilistic approach to WLAN user location estimation," *International Journal of Wireless Information Networks*, vol. 9, no. 3, pp. 155–164, 2002.

[25] F. Yu, M. H. Jiang, J. Liang et al., "An indoor localization of WiFi based on support vector machines," *Advanced Materials Research*, vol. 926-930, pp. 2438–2441, 2014.

[26] C. Feng, W. S. A. Au, S. Valaee, and Z. Tan, "Received-signal-strength-based indoor positioning using compressive sensing," *IEEE Transactions on Mobile Computing*, vol. 11, no. 12, pp. 1983–1993, 2012.

[27] X. Lu, C. Yu, H. Zou, H. Jiang, and L. Xie, "Extreme learning machine with dead zone and its application to WiFi based indoor positioning," in *Proceedings of the 13th International Conference on Control Automation Robotics and Vision (ICARCV '14)*, December 2014.

[28] B. Li, Y. Wang, H. K. Lee, A. G. Dempster, and C. Rizos, "Method for yielding a database of location fingerprints in WLAN," *IEE Proceedings—Communications*, vol. 152, no. 5, pp. 580–586, 2005.

[29] S. J. Pan, J. T. Kwok, Q. Yang, and J. J. Pan, "Adaptive localization in a dynamic WiFi environment through multi-view learning," in *Proceedings of the 22nd National Conference on Artificial Intelligence*, vol. 2, pp. 1108–1113, AAAI Press, Vancouver, Canada, 2007.

[30] H. Zhao, B. Huang, and B. Jia, "Applying kriging interpolation for WiFi fingerprinting based indoor positioning systems," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '16)*, pp. 1–6, April 2016.

[31] L. Ma, J. Li, Y. Xu, and W. Meng, "Radio map recovery and noise reduction method for green WiFi indoor positioning system based on inexact augmented lagrange multiplier algorithm," in *Proceedings of the 58th IEEE Global Communications Conference (GLOBECOM '15)*, pp. 1–5, IEEE, San Diego, Calif, USA, December 2015.

[32] H.-H. Liu, C.-W. Liao, and W.-H. Lo, "The fast collection of radio fingerprint for WiFi-based indoor positioning system," in *Proceedings of the 11th EAI International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QSHINE '15)*, Taipei, Taiwan, August 2015.

[33] Y. Du, D. Yang, and C. Xiu, "A novel method for constructing a WIFI positioning system with efficient manpower," *Sensors*, vol. 15, no. 4, pp. 8358–8381, 2015.

[34] Y. Zhuang, Z. Syed, Y. Li, and N. El-Sheimy, "Evaluation of two WiFi positioning systems based on autonomous crowdsourcing of handheld devices for indoor navigation," *IEEE Transactions on Mobile Computing*, vol. 15, no. 8, pp. 1982–1995, 2016.

[35] P. Prasithsangaree, P. Krishnamurthy, and P. K. Chrysanthis, "On indoor position location with wireless lans," in *Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '02)*, pp. 720–724, Lisboa, Portugal, September 2002.

[36] B. Li, J. Salter, A. G. Dempster, and C. Rizos, "Indoor positioning techniques based on wireless LAN," in *Proceedings of the 1st IEEE International Conference on Wireless Broadband and Ultra Wideband Communications (AusWireless '06)*, pp. 130–136, IEEE, Sydney, Australia, March 2006.

[37] V. Honkavirta, T. Perälä, S. Ali-Löytty, and R. Piché, "A comparative survey of WLAN location fingerprinting methods," in *Proceedings of the 6th Workshop on Positioning, Navigation and Communication (WPNC '09)*, March 2009.

[38] D. C. Montgomery, *Design and Analysis of Experiments*, John Wiley & Sons, 2008.

[39] G. Ilie and C. N. Ciocoiu, "Application of fishbone diagram to determine the risk of an event with multiple causes," *Management Research and Practice*, vol. 2, pp. 1–20, 2010.

[40] L. Chen, S. Ali-Löytty, R. Piché, and L. Wu, "Mobile tracking in mixed line-of-sight/non-line-of-sight conditions: algorithm and theoretical lower bound," *Wireless Personal Communications*, vol. 65, no. 4, pp. 753–771, 2012.

[41] T. S. Rappaport, *Wireless Communications: Principles and Practice*, vol. 2, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1996.

[42] S. Y. Seidel and T. S. Rappaport, "914 MHz path loss prediction models for indoor wireless communications in multifloored buildings," *IEEE Transactions on Antennas and Propagation*, vol. 40, no. 2, pp. 207–217, 1992.

[43] I. Ahmed, S. Orfali, T. Khattab, and A. Mohamed, "Characterization of the indoor-outdoor radio propagation channel at 2.4 GHz," in *Proceedings of the IEEE GCC Conference and Exhibition (GCC '11)*, pp. 605–608, February 2011.

[44] D. B. Faria, "Modeling signal attenuation in IEEE 802.11 wireless LANs," Tech. Rep. TR-KP06-0118, Kiwi Project, Computer Science Department, Stanford University, Stanford, Calif, USA, 2005.