University of Cyprus

Department of Electrical and Computer Engineering

Localization and Tracking in Wireless Networks for Fault Tolerance and Device Diversity

Christos Laoudias

A Dissertation Submitted to the University of Cyprus in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

March, 2014

VALIDATION PAGE

Christos Laoudias

Localization and Tracking in Wireless Networks for

Fault Tolerance and Device Diversity

The present Doctorate Dissertation was submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy at the Department of Electrical and Computer Engineering, and was approved on March 28, 2014 by the members of the Examination Committee.

Committee Chair	Dr. Marios M. Polycarpou
Research Supervisor	Dr. Christos G. Panayiotou
Committee Member	Dr. Christoforos Hadjicostis
Committee Member	Dr. Andreas Pitsillides
Committee Member	Dr. Federica Pascucci

DECLARATION OF DOCTORAL CANDIDATE

The present doctoral dissertation was submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy of the University of Cyprus. It is a product of original work of my own, unless otherwise mentioned through references, notes, or any other statements.

Christos Laoudias

Abstract

Location awareness is the key enabler for a wide variety of services that are envisioned or have already been deployed in the mobile and pervasive computing era. These services span from safety-critical applications in sensor networks, including large area monitoring, event localization and target tracking, to assistive and commercial applications in wireless networks, such as user navigation, visitor guidance, asset tracking and forwarding of location-dependent advertisements to consumer devices (e.g., smartphones, tablets, laptops, etc.).

In this thesis, we focus on two types of wireless networks that have attracted attention for determining location, namely binary Wireless Sensor Networks (WSN) and Wireless Local Area Networks (WLAN). A binary WSN is a network featuring a large number of cheap sensor nodes, densely deployed inside the monitoring area, that can only detect the presence of an event or not. Such simple nodes are preferred in certain applications because they are generally easier to implement, are less prone to calibration errors and can save bandwidth and energy by communicating only a single bit of information. On the other hand, WLANs are nowadays ubiquitous and the relevant infrastructure, i.e., Access Points (AP), can be quite dense especially in metropolitan areas. Due to the capability to penetrate obstacles and concrete walls, as well as the availability on commercial mobile devices, WLANs have the potential to deliver highly reliable location information right where the highest demand is anticipated in the near future; in large indoor environments, such as shopping malls, museums, exhibition centres, conference venues, and airports.

Despite their increasing popularity, binary WSNs and WLANs bring unique research challenges in respect to localizing and tracking events, objects or individuals. WSNs are usually deployed in harsh environments and are envisioned to work unattended for prolonged periods of time, while performing their tasks without human intervention. Also, WLANs are changing dynamically because APs can be removed or relocated or new APs may be installed.

In this sense, binary WSNs and WLANs may suffer from faults, caused either unintentionally or deliberately. For instance, in binary WSNs some nodes may fail unexpectedly due

to board overheating or battery depletion, reporting that they have detected an event/target when they did not and vice versa; or they may be compromised by a malicious attacker that aims to conceal the presence of an enemy vehicle. In WLAN-based localization, faults may appear in unpredictable ways, e.g., an AP may become unavailable as the result of power outage, hardware failure or due to network maintenance. Alternatively, faults may occur in case an adversary cuts the power supply of some APs or uses specialized equipment to severely jam the communication channels and make the attacked APs unavailable.

Another issue that is inherent in WLAN-based localization is device diversity. This is because users typically carry heterogeneous mobile devices that usually report measurements quite differently. Such devices may not be anticipated by the system, leading to significant localization errors. For instance, several studies report a linear relation between the signal strength received by two different devices at the same location. In that sense, device diversity can also be viewed as a bias sensor fault, where the sensor (i.e., WLAN adapter) in the device reports the disturbed, rather than the expected, measurements.

While hardware faults may severely degrade the localization and tracking accuracy in binary WSNs and WLANs, the use of heterogeneous devices affects user experience and hinders the proliferation of WLAN-based localization systems. These facts necessitate the development of appropriate fault models and fault tolerant algorithms, as well as device-insensitive localization algorithms to maintain a high level of accuracy in the presence of faults or when users request location information through diverse devices, respectively.

To this end, the main contributions of this thesis are threefold. First, we introduce fault models for binary WSNs and propose a multiple target tracking solution that exhibits high resilience to sensor node faults and is applicable to centralized and distributed network architectures. Second, we investigate fault models in the context of WLAN and propose fault detection mechanisms combined with fault tolerant localization algorithms to mitigate the effect of failed APs. Third, we look into device heterogeneity in WLAN-based localization and present two effective approaches to address this problem in real-time. We demonstrate the effectiveness of the proposed methods with extensive simulations and real-life experimental data.

Περίληψη

Η αντίληψη της θέσης αποτελεί το βασικό πυλώνα για μια μεγάλη ποικιλία υπηρεσιών που αναμένονται ή χρησιμοποιούνται ήδη στην εποχή του φορητού και διάχυτου υπολογισμού. Αυτές οι υπηρεσίες εκτείνονται από κρίσιμες εφαρμογές ασφάλειας σε δίκτυα αισθητήρων, όπως η παρακολούθηση μεγάλων εκτάσεων, ο προσδιορισμός της θέσης συμβάντων και η ιχνηλάτηση στόχων, μέχρι βοηθητικές και εμπορικές εφαρμογές σε ασύρματα δίκτυα, όπως ο προσδιορισμός της θέσης και η πλοήγηση των χρηστών, η ιχνηλάτηση περιουσιακών στοιχείων και η προώθηση διαφημίσεων, που σχετίζονται άμεσα με τη θέση των καταναλωτών, προς τις συσκευές τους (π.χ., έξυπνα κινητά τηλέφωνα, ταμπλέτες, φορητοί υπολογιστές, κτλ.).

Αυτή η διατριβή επικεντρώνεται σε δύο τύπους ασύρματων δικτύων τα οποία έχουν προσελκύσει την προσοχή για τον προσδιορισμό θέσης, ήτοι τα δυαδικά Ασύρματα Δίκτυα Αισθητήρων (ΑΔΑ) και τα Ασύρματα Τοπικά Δίκτυα (ΑΤΔ). Ένα δυαδικό ΑΔΑ είναι ένα δίκτυο που διαθέτει μεγάλο αριθμό φθηνών αισθητήρων, με πυκνή διάταξη εντός της περιοχής ελέγχου, οι οποίοι μπορούν να ανιχνεύσουν μόνο την παρουσία ή όχι ενός συμβάντος. Αυτοί οι απλοί αισθητήρες προτιμώνται σε ορισμένες εφαρμογές επειδή σε γενικές γραμμές είναι πιο εύκολα υλοποιήσιμοι, είναι λιγότερο επιρρεπείς σε λάθη βαθμονόμησης και έχουν τη δυνατότητα να εξοικονομήσουν εύρος ζώνης και ενέργεια με την αναμετάδοση πληροφορία ενός μόνο δυαδικού ψηφίου. Από την άλλη πλευρά, τα ΑΤΔ έχουν πλέον εξαπλωθεί παντού και η σχετική υποδομή, δηλ. τα Σημεία Πρόσβασης (ΣΠ), είναι αρκετά πυκνά ιδίως σε μητροπολιτικές περιοχές. Λόγω της ικανότητάς τους να διαπερνούν εμπόδια και συμπαγείς τοίχους, καθώς επίσης εξαιτίας της διαθεσιμότητάς τους σε εμπορικές φορητές συσκευές, τα ΑΤΔ έχουν την προοπτική να παρέχουν εξαιρετικά αξιόπιστες πληροφορίες τοποθεσίας εκεί όπου αναμένεται η υψηλότερη ζήτηση στο εγγύς μέλλον, δηλ. σε μεγάλους εσωτερικούς χώρους όπως εμπορικά κέντρα, μουσεία, εκθεσιακοί χώροι, συνεδριακά κέντρα και αεροδρόμια.

Παρά την αύξηση της δημοτικότητάς τους, τα δυαδικά ΑΔΑ και τα ΑΤΔ χαρακτηρίζονται από μοναδικές ερευνητικές προκλήσεις αναφορικά με τον προσδιορισμό της θέσης και την

ιχνηλάτηση συμβάντων, αντικείμενων ή ατόμων. Τα ΑΔΑ συνήθως αναπτύσσονται σε αντίξοες συνθήκες και προβλέπεται να λειτουργούν χωρίς επίβλεψη για παρατεταμένα χρονικά διαστήματα, ενώ εκτελούν τα καθήκοντά τους χωρίς ανθρώπινη παρέμβαση. Επίσης, τα ΑΤΔ αλλάζουν δυναμικά, επειδή μπορεί ορισμένα ΣΠ να αφαιρεθούν ή να μεταφερθούν σε άλλα σημεία ή ενδεχομένως να εγκατασταθούν νέα ΣΠ.

Υπό αυτό το πρίσμα, τα δυαδικά ΑΔΑ και τα ΑΤΔ ενδέχεται να υποφέρουν από σφάλματα, τα οποία προκαλούνται είτε ακούσια, είτε εσκεμμένα. Για παράδειγμα, σε δυαδικά ΑΔΑ ορισμένοι αισθητήρες μπορεί να παρουσιάσουν απροσδόκητη βλάβη εξαιτίας υπερθέρμανσης της πλακέτας ή εξάντλησης της μπαταρίας και να αναφέρουν ότι έχουν ανιχνεύσει ένα συμβάν/στόχο ενώ κάτι τέτοιο δεν ισχύει και το αντίστροφο. Ή μπορεί να έχει αλλοιωθεί η λειτουργία τους από έναν κακόβουλο εισβολέα, ο οποίος έχει ως στόχο να αποκρύψει την παρουσία ενός εχθρικού οχήματος. Όσον αφορά τον προσδιορισμό της θέσης σε ΑΤΔ, τα σφάλματα μπορούν να εμφανιστούν λόγω απρόβλεπτων καταστάσεων, π.χ., ένα ΣΠ μπορεί να μην είναι διαθέσιμο ως αποτέλεσμα της διακοπής ρεύματος, μιας βλάβης στο υλικό του, ή λόγω της συντήρησης του δικτύου. Εναλλακτικά, σφάλματα μπορούν να εμφανιστούν στην περίπτωση που ένας αντίπαλος διακόψει την τροφοδοσία ορισμένων ΣΠ ή χρησιμοποιήσει ειδικό εξοπλισμό για να δημιουργήσει σοβαρές παρεμβολές στα κανάλια επικοινωνίας ώστε να αποκόψει συγκεκριμένα ΣΠ.

Ένα άλλο ζήτημα, που είναι έμφυτο στον προσδιορισμό της θέσης με χρήση ΑΤΔ, είναι η ποικιλομορφία των συσκευών. Αυτό οφείλεται στο γεγονός ότι οι χρήστες χρησιμοποιούν συνήθως ετερογενείς φορητές συσκευές που συνήθως αναφέρουν τις μετρήσεις με αρκετά διαφορετικό τρόπο. Τέτοιες συσκευές ίσως να μην έχουν προβλεφθεί κατά τη λειτουργία του συστήματος, γεγονός που οδηγεί σε σημαντική απόκλιση της ακρίβειας. Για παράδειγμα, αρκετές μελέτες αναφέρουν μία γραμμική σχέση μεταξύ της ισχύος του σήματος που λαμβάνεται από δύο διαφορετικές συσκευές στην ίδια τοποθεσία. Υπό αυτή την έννοια, η ποικιλομορφία των συσκευών μπορεί επίσης να θεωρηθεί ως ένα σφάλμα πόλωσης, όπου ο αισθητήρας (δηλ. η κάρτα ΑΤΔ) στη συσκευή αναφέρει τις εσφαλμένες, αντί για τις αναμενόμενες μετρήσεις.

Ενώ τα σφάλματα στους αισθητήρες και τα ΣΠ μπορούν να υποβαθμίσουν σε σημαντικό βαθμό την ακρίβεια στον προσδιορισμό της θέσης και την ιχνηλάτηση σε δυαδικά ΑΔΑ και ΑΤΔ, η χρήση ετερογενών συσκευών επηρεάζει την εμπειρία του χρήστη και εμποδίζει τον πολλαπλασιασμό των συστημάτων προσδιορισμού της θέσης τα οποία βασίζονται σε ΑΤΔ. Για αυτό το λόγο είναι αναγκαία η κατάλληλη μοντελοποίηση των σφαλμάτων και η ανάπτυξη αλγόριθμων που είναι ανεκτικοί σε σφάλματα, καθώς και αλγόριθμων που δεν εξαρτώνται από το υλικό της συσκευής, ώστε να διατηρηθεί το υψηλό επίπεδο ακρίβειας στην περίπτωση που προκύψουν σφάλματα ή όταν οι χρήστες ζητήσουν πληροφορίες για την τοποθεσία τους χρησιμοποιώντας ετερογενείς συσκευές αντίστοιχα.

Για το σκοπό αυτό, οι κύριες συνεισφορές αυτής της διατριβής εντοπίζονται σε τρεις άξονες. Πρώτον, παρουσιάζουμε κατάλληλα μοντέλα σφαλμάτων για δυαδικά ΑΔΑ και προτείνουμε μία λύση για την ιχνηλάτηση πολλαπλών στόχων, η οποία χαρακτηρίζεται από υψηλή ανθεκτικότητα σε σφάλματα αισθητήρων και μπορεί να εφαρμοστεί τόσο σε κεντρικοποιημένες, όσο και σε κατανεμημένες αρχιτεκτονικές δικτύου. Δεύτερον, διερευνούμε μοντέλα σφαλμάτων στο πλαίσιο των ΑΤΔ και προτείνουμε μηχανισμούς ανίχνευσης σφαλμάτων σε συνδυασμό με αλγόριθμους προσδιορισμού της θέσης που είναι ανεκτικοί σε σφάλματα ώστε να μετριασθούν οι συνέπειες των ΣΠ που παρουσίασαν βλάβη. Τρίτον, εξετάζουμε τη χρήση ετερογενών συσκευών κατά τον προσδιορισμό της θέσης σε ΑΤΔ και παρουσιάζουμε δύο αποτελεσματικές προσεγγίσεις για την αντιμετώπιση αυτού του προβλήματος σε πραγματικό χρόνο.

Acknowledgements

First, I would like to express my deepest appreciation and gratitude to associate professor Christos G. Panayiotou; I see him more as my mentor and advisor, rather than my supervisor. I am grateful because he steered my research with his vision, while allowing me to take my own initiatives. I have always enjoyed playing basketball against him, as well as drinking beer with him along with the other members of our group, including Michalis P. Michaelides, Demetris Stavrou, Constantinos Heracleous and Theophanis Lambrou. I would like to thank them for our fruitful discussions all these years.

I would also like to thank all researchers at KIOS Research Center for being part of this inspiring environment that made my working time a real fun; especially, Demetrios G. Eliades, Michalis Skitsas, George Milis and Panayiotis Kolios with whom we cooperate in various tasks and projects. I am also grateful to the KIOS administrative personnel, Skevi Chrysanthou, Despina Petrou and Kalina Georgiades, for their help on the small things that made my life as a Ph.D. student easier.

In addition, I would like to express my thanks to the faculty of the Electrical and Computer Engineering Department, especially Marios Polycarpou and George Ellinas, for their suggestions and recommendations on my research. I would also like to thank Christoforos Chadjicostis, Andreas Pitsillides and Federica Pascucci for serving as my Ph.D. committee members and providing valuable feedback and comments on my thesis.

I would like to express my warmest thanks to Demetrios Zeinalipour-Yazti for our collaboration on joint research activities, as well as his team at DMSL for turning some of my research developments into real-life applications.

During my studies I had the opportunity to work together with professor Robert Piché at Tampere University of Technology, who I would like to thank for our ongoing collaboration. I would also like to thank the VTT Research Centre in Finland, especially Seppo Horsmanheimo and Paul Kemppi, for their hospitality during my visit that gave me the chance to gain hands-on experience on indoor localization systems at the early stages of my research.

I would like to acknowledge the Research Promotion Foundation of Cyprus and the University of Cyprus for supporting and funding my research.

This thesis is dedicated to my wife Maria for her love and patience, to my daughter Agni for giving me extra motivation and inspiration when she was born, as well as to my mother Agni and my father Ioannis for their endless support and understanding.

Publications

Journal articles

1. **C. Laoudias**, M. P. Michaelides, C. G. Panayiotou, ftTRACK: Fault Tolerant Target Tracking in Binary Sensor Networks, ACM Transactions on Sensor Networks, 2014. (in press)

2. M. P. Michaelides, **C. Laoudias**, C. G. Panayiotou, Fault Tolerant Localization and Tracking of Multiple Sources in WSNs using Binary Data, IEEE Transactions on Mobile Computing, 2014. (in press)

3. D. Zeinalipour-Yazti, **C. Laoudias**, C. Costa, M. Vlachos, M. I. Andreou, D. Gunopulos, Crowdsourced Trajectory Similarity with Smartphones, IEEE Transactions on Knowledge and Data Engineering, 25(6), pp. 1240–1253, 2013.

4. **C. Laoudias**, R. Piché, C. G. Panayiotou, Device Self-Calibration in Location Systems using Signal Strength Histograms, Journal of Location Based Services, 7(3), pp. 165–181, 2013.

5. **C. Laoudias**, M. P. Michaelides, C. G. Panayiotou, Fault Detection and Mitigation in WLAN RSS Fingerprint-based Positioning, Journal of Location Based Services, 6(2), pp. 101–116, 2012.

6. **C. Laoudias**, C. Panayiotou, C. Desiniotis, J. G. Markoulidakis, J. Pajunen, S. Nousiainen, Part One: The Statistical Terminal Assisted Mobile Positioning Methodology and Architecture, Computer Communications, 31(6), pp. 1126–1137, 2008.

Magazine articles

1. **C. Laoudias**, G. Larkou, D. Zeinalipour-Yazti, C. G. Panayiotou, Airplace: Indoor Geolocation on Smartphones Through WiFi Fingerprinting, ERCIM News, 2013(93), 2013.

2. G. Chatzimilioudis, A. Konstantinidis, **C. Laoudias** and D. Zeinalipour-Yazti, Crowdsourcing with Smartphones, IEEE Internet Computing, 16(5), pp. 36–44, 2012.

Refereed Conference/Workshop/Demo papers

1. P. Kolios, **C. Laoudias**, C. G. Panayiotou, Improving the Reliability of Emergency Response Networks using RevolverNet, 11th Workshop on Positioning, Navigation and Communication (WPNC), 2014.

2. L. Petrou, G. Larkou, **C. Laoudias**, D. Zeinalipour-Yazti, C. G. Panayiotou, Demonstration Abstract: Crowdsourced Indoor Localization and Navigation with Anyplace, 13th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), 2014.

3. **C. Laoudias**, D. Zeinalipour-Yazti, C. G. Panayiotou, Crowdsourced Indoor Localization for Diverse Devices through Radiomap Fusion, International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2013.

4. R. U. Haider, **C. Laoudias**, C. G. Panayiotou, Health Monitoring of WLAN Localization Infrastructure using Smartphone Inertial Sensors, International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2013.

5. **C. Laoudias**, M. P. Michaelides, C. G. Panayiotou, Sensor Health State Estimation for Target Tracking with Binary Sensor Networks, IEEE International Conference on Communications (ICC), 2013, pp. 1878–1882.

6. **C. Laoudias**, M. P. Michaelides, C. G. Panayiotou, Fault Tolerant Target Localization and Tracking in Binary WSNs using Sensor Health State Estimation, IEEE International Conference on Communications (ICC), 2013, pp. 1469–1473.

7. C.-L. Li, **C. Laoudias**, G. Larkou, Y.-K. Tsai, D. Zeinalipour-Yazti, C. G. Panayiotou, Demo: Indoor Geolocation on Multi-Sensor Smartphones, 11th ACM International Conference on Mobile Systems, Applications and Services (MobiSys), 2013, pp. 503–504.

8. **C. Laoudias**, R. Piché, C. Panayiotou, Device Signal Strength Self-Calibration using Histograms, International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2012, pp. 1–8.

9. M. Raspopoulos, **C. Laoudias**, L. Kanaris, A. Kokkinis, C. G. Panayiotou, S. Stavrou, Cross Device Fingerprint-based Positioning using 3D Ray Tracing, 8th International Wireless Communications and Mobile Computing Conference (IWCMC), 2012, pp. 147–152.

10. A. Konstantinidis, G. Chatzimilioudis, **C. Laoudias**, S. Nicolaou and D. Zeinalipour-Yazti, Towards Planet-Scale Localization on Smartphones with a Partial Radiomap, 4th ACM International Workshop on Hot Topics in Planet-Scale Measurement (HotPlanet), 2012, pp. 9–14.

11. M. Raspopoulos, **C. Laoudias**, L. Kanaris, A. Kokkinis, C. G. Panayiotou, S. Stavrou, 3D Ray Tracing for Device-Independent Fingerprint-based Positioning in WLANs, 9th Workshop on Positioning, Navigation and Communication (WPNC), 2012, pp. 109–113.

12. **C. Laoudias**, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, C. G. Panayiotou, Demo: The Airplace Indoor Positioning Platform, 10th ACM International Conference on Mobile Systems, Applications and Services (MobiSys), 2012, pp. 467–468.

13. **C. Laoudias**, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, C. G. Panayiotou, The Airplace Indoor Positioning Platform for Android Smartphones, Demo at 13th IEEE International Conference on Mobile Data Management (MDM), 2012, pp. 312–315. **Best Demo Award**

14. C. Laoudias, M. P. Michaelides, C. G. Panayiotou, Fault Detection and Mitigation in WLAN RSS Fingerprint-based Positioning, International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011, pp. 1–7.

15. M. P. Michaelides, **C. Laoudias**, C. G. Panayiotou, Fault Tolerant Target Localization and Tracking in Wireless Sensor Networks Using Binary Data, IEEE Global Telecommunications Conference (GLOBECOM), 2011, pp. 1–6.

16. F. Della Rosa, L. Xu, J. Nurmi, M. Pelosi, **C. Laoudias**, A. Terrezza, Hand-Grip and Body-Loss Impact on RSS Measurements for Localization of Mass Market Devices, International Conference on Localization and GNSS (ICL-GNSS), 2011, pp. 58–63.

17. D. Zeinalipour-Yazti, **C. Laoudias**, M. I. Andreou, D. Gunopulos, Disclosure-free GPS Trace Search in Smartphone Networks, 12th International Conference on Mobile Data Management (MDM), 2011, vol.1, pp. 78–87.

18. **C. Laoudias**, M. P. Michaelides, C. G. Panayiotou, Fault Tolerant Fingerprint-based Positioning, IEEE International Conference on Communications (ICC), 2011, pp. 1–5.

19. C. Costa, **C. Laoudias**, D. Zeinalipour-Yazti, D. Gunopulos, SmartTrace: Distributed Trajectory Search and Retrieval in Smartphone Networks, Demo at 27th IEEE International Conference on Data Engineering (ICDE), 2011, pp. 1288–1291.

20. **C. Laoudias**, M. P. Michaelides, C. G. Panayiotou, Fault Tolerant Positioning using WLAN Signal Strength Fingerprints, International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2010, pp. 1–8.

21. **C. Laoudias**, C. G. Panayiotou, P. Kemppi, On the RBF-based Positioning using WLAN Signal Strength Fingerprints, 7th Workshop on Positioning, Navigation and Communication (WPNC), 2010, pp. 93–98.

22. M. P. Michaelides, **C. Laoudias**, C. G. Panayiotou, Fault Tolerant Detection and Tracking of Multiple Sources in WSNs using Binary Data, 48th IEEE Conference on Decision and Control (CDC), 2009, pp. 3769–3774.

23. **C. Laoudias**, P. Kemppi, C. Panayiotou, Localization using Radial Basis Function Networks and Signal Strength Fingerprints in WLAN, IEEE Global Telecommunications Conference (GLOBECOM), 2009, pp. 1–6.

24. **C. Laoudias**, D. Eliades, P. Kemppi, C. Panayiotou, M. Polycarpou, Indoor Localization using Neural Networks with Location Fingerprints, LNCS Artificial Neural Networks - ICANN 2009, pp. 954–963.

25. **C. Laoudias**, C. Desiniotis , J. Pajunen, S. Nousiainen, C. Panayiotou, J. G. Markoulidakis, Ubiquitous Terminal Assisted Positioning Prototype, IEEE Wireless Communications and Networking Conference (WCNC), 2008, pp. 3261–3266.

Under preparation

1. **C. Laoudias**, D. Zeinalipour-Yazti, P. Kolios, C. G. Panayiotou, Device Independent Localization Using Mean Differential Fingerprinting, IEEE Transactions on Mobile Computing, 2014.

2. L. Petrou, G. Larkou, **C. Laoudias**, D. Zeinalipour-Yazti, C. G. Panayiotou, The Anyplace Indoor Information Service, IEEE Internet Computing, 2014.

Other Workshop/Demo papers

1. **C. Laoudias**, M. P. Michaelides, C. G. Panayiotou, Fault Tolerant Target Tracking in Binary Wireless Sensor Networks, 6th Cyprus Workshop on Signal Processing and Informatics (CWSPI), 2013.

2. L. Petrou, G. Larkou, **C. Laoudias**, D. Zeinalipour-Yazti and C. G. Panayiotou, Anyplace: Indoor Positioning and Navigation in the Big-Data Era, Demo at International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2013.

3. **C. Laoudias**, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, C. G. Panayiotou, The Airplace Indoor Positioning System, 5th Cyprus Workshop on Signal Processing and Informatics (CWSPI), 2012.

4. C.-L. Li, **C. Laoudias**, G. Larkou, G. Chatzimilioudis, D. Zeinalipour-Yazti, C. G. Panayiotou, Hybrid Indoor Positioning on Multi-Sensor Android Smartphones, Demo at International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2012.

5. **C. Laoudias**, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, C. G. Panayiotou, A Platform for the Evaluation of Fingerprint Positioning Algorithms on Android Smartphones, Demo at International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011.

6. **C. Laoudias**, C. Costa, D. Zeinalipour-Yazti, C. G. Panayiotou, An Indoor Trajectory Comparison Framework for Android Smartphones, Demo at International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2011.

7. C. Costa, **C. Laoudias**, D. Zeinalipour-Yazti, D. Gunopulos, SmartTrace: Finding Similar Trajectories in Smartphone Networks without Disclosing the Traces, Demo at 10th Hellenic Data Management Symposium (HDMS), 2011.

8. **C. Laoudias**, C. G. Panayiotou, Indoor Positioning in WLAN using Radial Basis Function Networks with Received Signal Strength Fingerprints, 3rd Cyprus Workshop on Signal Processing and Informatics (CWSPI), 2010.

9. C. Costa, C. Christoforou, G. Nicolaides, D. Papadiomidous, **C. Laoudias**, D. Zeinalipour-Yazti, A Distributed Spatio-Temporal Similarity Search Framework using Android-based Smartphones, Demo at 9th Hellenic Data Management Symposium (HDMS), 2010.

10. **C. Laoudias**, C. Panayiotou, J. Markoulidakis, C. Desiniotis, Simulation Analysis on the Efficiency of STAMP Method, LIAISON-ISHTAR Workshop on LBS Trends and R&D, 2006.

Contents

Nomenclature

$(\cdot)^T$	Vector transpose operator
$\alpha\%$	Percentage of faulty sensors nodes in the field or faulty APs in the WLAN setup
$\bar{\bar{\mathbf{R}}}$	MDF differential radiomap
$\bar{\bar{\mathbf{r}}}_i$	MDF reference fingerprint at location ℓ_i
$\bar{\bar{\mathbf{s}}}$	MDF localization fingerprint
$\bar{\bar{d}}_i^2$	MDF squared Euclidean distance between the observed MDF fingerprint during localization $\bar{\bar{\mathbf{s}}}$ and the MDF reference fingerprint $\bar{\bar{\mathbf{r}}}_i$
$\bar{\bar{r}}_{ij}$	MDF difference between the j-th AP and $\bar{r}_i$ in the reference fingerprint $\bar{\bar{\mathbf{r}}}_i$ at location ℓ_i
$\bar{\bar{s}}_j$	MDF difference between the j-th AP and $\bar{s}$ in the localization fingerprint $\bar{\bar{\mathbf{s}}}$
$\bar{\epsilon}$	Mean positioning error for a single route in a WLAN setup
$\bar{r}_i$	Mean RSS value over all APs in the RSS reference fingerprint $\mathbf{r}_i$ at location ℓ_i
$\bar{s}$	Mean RSS value over all APs in the localization fingerprint $\mathbf{s}$
$\bar{X}^i(t)$	Position components in the state $X^i(t)$ of the i-th particle at time t
β_L	Scaling factor for the estimation rule of the SNAP localization error
$\check{\mathbf{R}}$	SSD differential radiomap
$\check{\mathbf{r}}_i$	SSD reference fingerprint at location ℓ_i
$\check{\mathbf{s}}$	SSD fingerprint during localization
$\check{d}_i^2$	SSD squared Euclidean distance between the observed SSD fingerprint $\check{\mathbf{s}}$ during localization and the SSD reference fingerprint $\check{\mathbf{r}}_i$

$\check{r}_{ij}$	SSD difference between the j-th AP and the anchor AP ρ in the reference fingerprint $\check{\mathbf{r}}_i$ at location ℓ_i
$\check{s}_j$	SSD difference between the j-th AP and the anchor AP ρ in the localization fingerprint $\check{\mathbf{s}}$
$\delta(\cdot)$	Dirac delta function
ℓ	Unknown user location in WLAN fingerprinting
ℓ_i	i-th reference location in a WLAN setup
$\ell_k(t)$	Location of the k-th moving source at time t
ℓ_n	Location of the n-th static sensor node with coordinates (x_n, y_n)
$\ell_s(t)$	Location $(x_s(t), y_s(t))$ of a single moving source at time t
γ	Fault detection threshold in RSS fingerprinting
$\hat{\ell}_s(t)$	Estimated source location $(\hat{x}_s(t), \hat{y}_s(t))$ at time t
$\hat{\pi}_n(t)$	Estimated state probabilities vector of sensor node n at time t
$\hat{\pi}_n^i(t)$	Estimated probability of sensor node n being at state i, $i \in \{F, H\}$
$\hat{C}_n(t)$	Estimated state transition matrix of the HMM health state estimator for sensor n at time t
$\hat{e}_s(t)$	Uncertainty associated with the estimated source location $\hat{\ell}_s(t)$
$\hat{p}_n^{i,j}(t)$	Estimated MC transition probability from state i to state j with $i, j \in \{F, H\}$
$\hat{s}_n(t)$	Estimated health state of sensor n at time t
$\mathbf{E}[\ell \mid \mathrm{s}]$	Expected value of the user location ℓ given the localization fingerprint s
$\mathbf{R}$	Crowdsourced RSS fingerprint radiomap
$\mathbf{R}^{(m)}$	RSS fingerprint radiomap collected with device $D^{(m)}$
$\mathbf{r}_i$	Reference fingerprint associated with location ℓ_i
$\mathbf{r}_i^{(m)}$	RSS reference fingerprint collected with device $D^{(m)}$ at location ℓ_i

$\mathbf{s}$	Localization fingerprint measured at the unknown user location ℓ
$\mathbf{S}^{(m)}$	Vector containing the minimum $s_{min}^{(m)}$ and $s_{max}^{(m)}$ maximum RSS values in the histograms of device $D^{(m)}$
$\mathcal{A}$	Rectangular sensor field
$\mathcal{E}$	Mean positioning error
$\mathcal{E}_s$	Cumulative sensor health state estimation error
$\mathcal{E}_T$	Tracking error pertaining to the whole target path
$\mathcal{L}_l$	Pseudo-likelihood matrix for leader node l in the dSNAP localization algorithm
$\mathcal{P}_Y^{sim}$	Probability of correct location estimation using simulations
$\mathcal{P}_C$	Analytical probability of correct location estimation using RSS fingerprints
$\mathcal{P}_Q$	Analytical probability of correct location estimation using MDF fingerprints
$\mathcal{P}_R$	Analytical probability of correct location estimation using SSD fingerprints
$\omega^i(t)$	Weight of the i-th particle at time t
$\overline{ROC_m}$	Set of all grid cells that are covered by the ROC of sensor node m
$\Phi(x)$	CDF of a Gaussian random variable $\mathcal{N}(0,1)$, $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp(-\frac{y^2}{2})dy$
$\pi_n(t)$	State probabilities vector of sensor node n at time step t
π_n^i	Steady state probability of state $i \in \{RS, SA1, SA0, H\}$ for sensor node n
$\pi_n^i(t)$	Probability of sensor node n being at state $i,\ i \in \{RS, SA1, SA0, H\}$
ρ	Anchor AP for creating the SSD differential fingerprints
$\sigma(t)$	Confidence in the measurement $Y(t)$ in the target mobility model at time t
σ^2	Variance of the noise in the RSS values
σ_w^2	Variance of the noise in the sensor node measurements
τ	Threshold when D_s is used as a test statistic for attack detection
$\tilde{\ell}_s(t)$	Filtered estimated source location $(\tilde{x}_s(t), \tilde{y}_s(t))$ at time t

$\tilde{\mathbf{R}}$	DIFF differential radiomap
$\tilde{\mathbf{r}}_i$	DIFF reference fingerprint at location ℓ_i
$\tilde{\mathbf{s}}$	DIFF fingerprint during localization
$\tilde{d}_i^2$	Squared Euclidean distance between the DIFF observed fingerprint during localization $\tilde{\mathbf{s}}$ and the DIFF reference fingerprint $\tilde{\mathbf{r}}_i$
$\tilde{d}_n(t)$	Radial distance between sensor node n and the filtered estimated source location $\tilde{\ell}_s(t)$ at time t
$\tilde{r}_n(t)$	Estimated binary error signal for sensor node n at time t
$\tilde{r}_{ijk}$	DIFF difference between the j-th and k-th APs in the DIFF reference fingerprint $\tilde{\mathbf{r}}_i$ at location ℓ_i
$\tilde{s}_{jk}$	DIFF difference between the j-th and k-th APs in the DIFF localization fingerprint $\tilde{\mathbf{s}}$
$\widehat{\ell}$	User location estimated with RSS fingerprinting
ζ	Distance attenuation power
$A_n(t)$	Alarm status of the n-th sensor node at time t
A_r	Range of RSS values measured in WLAN setup with a device
A_{12}	Slope parameter of the linear fitting between the RSS values reported by devices $D^{(1)}$ and $D^{(2)}$
b_s	Buffer size of the self-calibration approach
B_{12}	Offset parameter of the linear fitting between the RSS values reported by devices $D^{(1)}$ and $D^{(2)}$
C	State transition matrix of the MC fault model
c	Constant signal energy generated by a single moving source
c'	Anticipated source signal energy within ftTRACK
c_k	Constant signal energy generated by the k-th moving source

d	Euclidean distance in 2-D physical space
$D^{(m)}$	Mobile device $m = 1, \ldots, M$ used for localization in WLAN
d_i	Euclidean distance between the observed fingerprint during localization $\mathbf{s}$ and the reference fingerprints $\mathbf{r}_i$ in the radiomap
$d_n(t)$	Radial distance between sensor node n and the source at time t
D_s	Minimum distance between the localization fingerprint and the fingerprints in the radiomap
$d_{n,k}(t)$	Radial distance between sensor node n and the k-th source at time t
$D_{sum}^{(K)}$	Sum of distances to the K nearest neighbours in RSS fingerprinting
F	Faulty sensor node state
f	Path loss exponent
$F(x)$	CDF that gives the probability of observing an RSS value that is less than x
$F^{-1}(y)$	Inverse CDF that returns the RSS value corresponding to the y-th CDF percentile
$F_1(x)$	Empirical CDF of the reference device $D^{(1)}$
$F_2(x)$	Empirical CDF of the user device $D^{(2)}$
F_n	Detection function in the leader election protocol
H	Healthy sensor node state
$h(x)$	Strictly increasing waiting period function
I_n	$n \times n$ identity matrix
K	RSS value at reference distance $d = 1$ m
L	Set of predefined reference location in WLAN fingerprinting
l	Number of predefined reference locations in a WLAN setup
$L^{(m)}$	Subset of the reference locations visited by device $D^{(m)}$

$L_{max}(t)$	Maximum value in the $\mathcal{L}(t)$ matrix at time t
LV_i	Likelihood value of the user being located at location ℓ_i
M_i	Number of devices with $1 \leq M_i \leq M$ that contribute RSS data collected at location ℓ_i
N	Number of static sensor nodes in the field
n	Number of APs in a WLAN setup
N_p	Number of particles in the particle filter
N_z	Number of zones for dividing the range of RSS values in a WLAN setup
n_{miss}	Number of missing WLAN APs during localization
$p(\ell_i)$	Prior probability of reference location ℓ_i
$p(\mathbf{s} \mid \ell_i)$	Likelihood of the observed fingerprint **s** given the reference location ℓ_i
$p(\mathbf{s})$	Probability of observing fingerprint **s** in the entire localization area
$p(s_j \mid \ell_i)$	Probability of observing the RSS value s_j from the j-th AP at location ℓ_i
$p^{i,j}$	MC transition probability from state i to state j with $i, j \in \{RS, SA1, SA0, H\}$
P_f	Probability of sensor node fault in the random fault model
$p_n^f(t)$	Probability of sensor node n having *wrong* output at time t given that its state is *Faulty*
$p_n^h(t)$	Probability of sensor node n having *wrong* output at time t given that its state is *Healthy*
P_{fa}	Probability of false alarms
P_{max}	Maximum likelihood of the candidate locations ℓ_i
P_{nd}	Probability of no detection
P_{sum}	Negative logarithm of the sum of likelihoods of the candidate locations ℓ_i
$Q(x)$	Right-tail probability of a Gaussian random variable $\mathcal{N}(0,1)$, $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \exp(-\frac{y^2}{2}) dy$

R_C	Radius of the circular ROC_n centred at the location of the n-th sensor node
R_I	Radius of the circular ROI centred at the source location
R_i	Set of APs that are present in fingerprint $\mathbf{r}_i$
$r_n(t)$	Binary error signal for sensor node n at time t
$R_n^{i,j}(t)$	Counter incrementing by one when $\hat{s}_n(t-1) = i$ is followed by $\hat{s}_n(t) = j$
R_S	Communication range for all sensor nodes
R_{cd}	Correct detections rate
R_{fd}	False detections rate
r_{ij}	RSS value at location ℓ_i related to the j-th AP
$r_{ij}^{(m)}$	RSS value from the j-th AP collected using device $D^{(m)}$ at the reference location ℓ_i
r_{sep}	Separation distance between two moving sources
RoC_j	Region of Coverage of the j-th AP in a WLAN setup
ROC_n	Region of Coverage of sensor node n
ROI	Region of Influence of a moving source
ROS_n	Region of Subscription of sensor node n
S	Set of APs that are present in fingerprint $\mathbf{s}$
s_j	RSS value at the unknown user location ℓ related to the j-th AP
s_{max}	Maximum RSS value measured by a device in a WLAN setup
s_{min}	Minimum RSS value measured by a device in a WLAN setup
$s_{n,k}(t)$	Signal from the k-th moving source received at the n-th sensor node at time t
T	Detection threshold of binary sensor nodes
t	Time step
T_s	Sampling interval

$U(t)$	Gaussian measurement noise in the source mobility model
$W(t)$	Gaussian process noise in the source mobility model
$w_n(t)$	Gaussian noise disturbing the measurements of the n-th sensor at time t, $w_n(t) \sim \mathcal{N}(0, \sigma_w^2)$
X	Gaussian noise disturbing the RSS values, $X \sim \mathcal{N}(0, \sigma^2)$
$X(t)$	Process state for moving source at time t
$X^i(t)$	State of the i-th particle at time t
$Y(t)$	Source location measurement vector at time t
Z_m	m-th subrange (interval) of RSS values
$z_n(t)$	Signal received due to a moving source at sensor node n at time t
ZoC_{mj}	Zone of Coverage of the j-th AP in the RSS subrange Z_m
$\mathcal{G}$	Grid over the sensor field with dimensions $R_x \times R_y$ and grid resolution g
$\mathcal{L}(t)$	Pseudo-likelihood matrix for SNAP localization algorithm at time t
AP	Access Point in a WLAN setup
CDF	Cumulative Distribution Function
CE	Centroid Estimator
D-FTLEP	Distributed Fault Tolerant Leader Election Protocol
D-NLEP	Distributed Naive Leader Election Protocol
DIFF	Differential signal strength using the RSS differences between pairwise AP combinations
DS	Device Specific radiomap approach
dSNAP	Distributed Subtract on Negative Add on Positive localization algorithm
eCDF	Empirical CDF
EM	Expectation-Maximization

FTML	Fault Tolerant Maximum Likelihood
ftTRACK	Fault tolerant target tracking architecture
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
GRNN	Generalized Regression Neural Network
H-KNN	Hybrid KNN algorithm
H-MED	Hybrid MED algorithm
H-MMSE	Hybrid MMSE algorithm
HLF	Hyperbolic Location Fingerprinting
HMM	Hidden Markov Model
KNN	K-Nearest Neighbour localization algorithm
LVDF	Local Vote Decision Fusion approach
MC	Markov Chain
MDF	Mean Differential Fingerprint approach
MED	KNN algorithm variant that uses a median-based distance metric
ML	Maximum Likelihood
MLP	Multi Layer Perceptron
MMSE	Minimum Mean Square Error localization algorithm
NC	No Calibration approach
NLOS	Non-Line-Of-Sight
NN	Nearest Neighbour
PDF	Probability Density Function
PDR	Pedestrian Dead Reckoning
PF	Particle Filter

RBF	Ranked-Based Fingerprinting
RBFN	Radial Basis Function Network
RS	Reverse Status sensor node fault
RSS	Received Signal Strength
SA0	Stuck-At-0 sensor node fault
SA1	Stuck-At-1 sensor node fault
SC	Self-calibration approach
SCmed	Modified self-calibration approach that uses a median-based offset estimator
SNAP	Subtract on Negative Add on Positive localization algorithm
SNAPft-z	Fault tolerant variant of the SNAPz algorithm for WLAN localization
SNAPz	Modified SNAP algorithm that uses RSS zones for WLAN localization
SSD	Signal Strength Difference using an anchor AP
SVR	Support Vector Regressor
TI	Trust Index
WLAN	Wireless Local Area Network
WSN	Wireless Sensor Network

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Importance of Location Information

Location has nowadays become an integral part of the mobile user's daily life. For example, Google is already reporting that a third of searches on portable electronic devices refer to spatially relevant content, while 94% of smartphone users have searched for local information. In particular, there has been an increasing interest in location-aware content specific to large indoor environments, such as shopping malls, museums, exhibition centres and airports. This is explained by recent statistics from Strategy Analytics indicating that people spend 80-90% of their time inside buildings, while 70% of cellular calls and 80% of data connections originate from indoors. These facts have triggered the development of indoor mobile applications, such as location-based advertising, in-building guidance, indoor navigation, asset tracking, and others.

Furthermore, accurate and reliable location information is very important not only for supporting consumer services, but also for enabling security and safety-critical applications. These are typically deployed in harsh conditions or hostile environments and are usually built on top of sensors networks. Indicative applications include environmental monitoring, military area surveillance, safeguarding critical infrastructure, localizing an event (e.g., fire or pollutant source) and tracking a target (e.g., enemy vehicle or intruder).

1.2 Wireless Network Aided Localization

Admittedly, Global Navigation Satellite Systems (GNSS), such as the Global Positioning System (GPS), is the dominant technology for outdoor localization, tracking and navigation applications. GNSS deliver localization accuracy within few meters, however such systems have several shortcomings. For instance, GNSS receivers have high energy requirements, which is undesirable for battery-powered devices, such as sensor nodes. Moreover, satellite-based systems have limited availability in urban canyons and indoor environments due to the severe attenuation of the positioning signals. In addition, when the GNSS receiver is turned on it may take several minutes to detect and lock enough satellite signals to determine location (i.e., high time-to-first-fix). Finally, not all devices are equipped with GNSS chipsets, while several applications, such as the weather forecast in the user's home town, do not require the high accuracy provided by GNSS.

These facts have motivated the development of alternative localization systems that rely on wireless communication technologies, including Wireless Local Area Networks (WLAN) and Wireless Sensor Networks (WSN). Such systems are preferred because they either exploit the existing network infrastructure, i.e., Access Points (AP), or the network is cheap and easy to deploy like in the case of WSNs. Moreover, localization systems based on wireless networks are non-intrusive because they do not require the installation of dedicated and costly hardware (e.g., transmitters, antennas, cabling) or privacy-infringing equipment (e.g., cameras).

WSNs comprise a large number of cheap sensor nodes that are usually densely deployed inside the area of interest. Such nodes are envisioned to operate unattended for prolonged periods of time and are expected to perform their task successfully without human intervention. Event source localization is a popular task undertaken by WSNs, where the events can be intruders, enemy vehicles, pollutant sources or fires depending on the application [3]. In all situations, the event source emits a signal or substance that attenuates inside the area under observation and is measured by a subset of the nodes that are located in the vicinity of the event. The objective is to use the sensor readings to localize the event source, e.g., by translating the signals measured by the neighbouring sensors, whose location is known, into distances from the corresponding sensors.

In particular, binary sensors can only detect the presence of a source inside their sensing area or not. Such simple sensor nodes are attractive for certain applications because they are generally easier to implement and have low circuitry complexity, while they are also less sensitive to calibration mismatches and varying sensor sensitivities. Moreover, the bandwidth usage is reduced and valuable sensor energy is conserved because single-bit information is transmitted in case of detection. Based on the sensor binary observations the goal is to localize the source and track its movement in real-time.

Localization in WLAN can be performed using radio signal propagation models that relate the unknown location of the user-carried device to the known locations of the surrounding APs. Different types of measurements can be employed, including angle, timing and signal strength measurements. Subsequently, the user location can be determined by solving a system of non linear equations in a least squares sense. However, modelling signal propagation is difficult, especially in deep urban and indoor environments, due to the presence of non-line-of-sight (NLOS) paths between the receiver and the transmitter and the prevalence of multipath conditions. Thus, high localization errors may occur.

Fingerprinting localization methods address this issue by using fingerprints collected a priori in the entire area of interest. The fingerprints are vectors containing location related information, e.g., angle, timing or signal strength samples, that are associated with a set of predefined reference locations to build the radiomap of the area. Location can then be estimated by finding the best match between the observed fingerprint during localization and the reference fingerprints.

WLAN fingerprinting using Received Signal Strength (RSS) measurements has become an attractive solution, owing to the abundance of WLAN APs, especially in urban and indoor areas, combined with the proliferation of mobile devices with WLAN connectivity and the ease of collecting RSS values on mass market devices. Interestingly, the exact location of the APs is no longer required. This is important because the localization system can exploit not only the limited number of APs inside a *private* fully-controlled network, where AP locations can be easily obtained, but also public and residential APs like in the case of *public* localization systems that cover larger urban areas, e.g., *Google Maps Geolocation*[1] or *Skyhook*[2].

1.3 Thesis Motivation

Nowadays, WSNs can well handle difficult tasks in a collaborative fashion and are becoming increasingly popular for demanding applications, such as large area monitoring, event localization and target (source) tracking [148]. However, experience has shown that the currently available sensor nodes often fail due to various reasons, including battery depletion, calibration problems, software errors or harsh environmental conditions.

When binary WSNs are considered, often some nodes may fail and suffer from Byzantine faults, e.g., they may transmit that they have detected a target when they did not, or they may not transmit anything when a target is actually present. Such behaviours have been doc-

[1]The Google Maps Geolocation API, https://developers.google.com/maps/documentation/business/geolocation/

[2]Skyhook Inc., http://www.skyhookwireless.com

umented in the literature; for example, in some cases sensors were erroneously reporting the presence of an event due to board overheating or due to software bugs [6, 112]. Similarly, events were not detected due to wrong threshold configuration or were not reported due to dropped packets. In all these cases, network performance with respect to the intended task can be severely degraded. Therefore, fault tolerant methods are required.

Another reason that motivates our work is the fact that in such settings it is generally infeasible or costly to know the health state of every individual sensor node. Estimating the sensors' health state would be beneficial in order to treat the corresponding observations accordingly. For instance, measurements originating from sensors that are considered as faulty could be ignored or filtered before processing.

In WLAN fingerprinting systems, the focus so far has been on reducing the localization error. While location accuracy is an important requirement, fault tolerance is also highly desirable because the RSS values in the fingerprints may be corrupted in the presence of faults, thus leading to accuracy degradation. For instance, some WLAN APs may become unavailable during positioning, either due to unexpected failures, such as power outages, or as a result of a malicious attack. Even though fault tolerance is important, surprisingly it has received little attention in the literature. Our goal is to develop localization methods that are resilient to AP faults.

Even if no AP faults occur, the reliability of location estimates in WLAN fingerprinting systems also depends on the user device itself. This is because users usually carry heterogeneous devices that do not report the RSS measurements from the surrounding APs in the same way, e.g., due to different WLAN chipsets, varying antenna gain, etc. In most real-life applications it is expected that the user-carried device will be different from the reference device, which was employed to collect the RSS radiomap, thus incurring higher localization errors compared to the case of using the same device. A linear relation between RSS values measured by diverse devices has been reported in several experimental studies [52, 88, 116]. Thus, the device heterogeneity problem is equivalent to using the same reference device for localization, while the on-board WLAN adapter suffers from a bias fault that disturbs the expected RSS values. Our objective is to develop localization methods that are robust to device heterogeneity.

1.4 Research Challenges

1.4.1 Localization and tracking in binary WSNs

In the context of binary WSNs, several algorithms have been presented for target localization which differ in complexity, accuracy and fault tolerance [38, 103, 108, 113]. These algorithms, however, are snapshot estimators that do not consider possible correlations among sensor observations while a target is moving inside the field. In target tracking applications, the data received by the sensor nodes will appear correlated both in space and time [139]. In particular, spatial correlation will be observed between sensors located in the vicinity of the target inside an area that is based on the signal propagation characteristics. For example, assuming a uniform propagation model, we expect the sensor nodes located inside a disc centred around the target location to detect the target with high probability. On the other hand, temporal correlation is a direct consequence of the constraints imposed by the target movement. For example, a binary sensor node is expected to detect a target for several consecutive measurement periods, while it is passing through the sensor's neighbourhood. Such spatiotemporal information is used for localizing and tracking a target. Several tracking algorithms rely on Bayesian filtering, including Kalman and particle filters [7, 26, 28, 60, 114]. However, these approaches do not consider sensor faults.

The spatiotemporal information provided by the sensors is not only useful for localization and tracking. Additionally, it can assist in detecting sensor faults. It is often the case that certain types of sensor faults appear to be highly correlated in both the time and space domain. Consider, for example a sensor being stuck at a particular value because of overheating, battery depletion or software malfunction [6, 112]. During this fault, the sensor would produce several consecutive erroneous values that appear highly correlated in the time domain. As another example, consider an adversary who deploys a malicious actuator network in order to perturb the sensor readings in the actuators' neighbourhoods [37]. Under such conditions, the sensor faults will also exhibit spatial correlation based on the positions of the actuator nodes. Therefore, the challenge is to exploit the spatiotemporal information, obtained while a target is moving through the field, to distinguish between healthy and faulty sensors in order to employ only those sensors that are thought to be healthy for the localization and tracking tasks.

1.4.2 Localization in WLAN

Several localization systems rely on WLANs and exploit RSS measurements from the surrounding APs to determine the unknown user location through fingerprinting. Location accuracy is an important requirement and has been the main interest of researchers so far. However, fault tolerance is also highly desirable because the RSS values in the fingerprints may be corrupted in the presence of faults, thus leading to accuracy degradation. For instance, some APs may become unavailable during positioning either due to unexpected failures, such as power outages, or as a result of a malicious attack. In this context, there are three research challenges that need to be addressed: fault modelling and simulation, fault detection and fault tolerant localization.

Existing fault models consider only the case that the RSS values are altered by a constant value, so that the expected RSS value is either attenuated or amplified, or disturbed by Gaussian noise. Although these models capture some hardware malfunctions and attacks, they fail to capture other realistic scenarios, such as APs being removed or relocated, that manifest themselves as faults from the perspective of the fingerprinting system.

Moreover, detecting faulty APs is important in order to trigger an alert for maintenance or security personnel and switch to a fault tolerant positioning method, if required. However, fault detection has received little attention so far.

Lastly, traditional fingerprinting approaches do not consider faults. Thus, the challenge is to build fault tolerant algorithms to guarantee that the localization error is acceptable in the presence of faults.

1.4.3 Localization for Diverse Devices

In fingerprinting systems the best localization accuracy is guaranteed in case the user carries the *same* device that was used to collect the data for the RSS radiomap. The existence of a wide variety of WLAN-enabled mobile devices, which may report the observed RSS values in a different way, necessitates a *calibration* step to make the user-carried device compatible with the existing radiomap. Several calibration methods rely on data fitting to create a mapping between the RSS values collected with different devices [52,62,109,116]. However, these methods require the collection of a considerable volume of data at several *known* or *unknown* locations prior to positioning, which can be prohibitive in real-life applications.

Alternatively, calibration-free methods remove the device-dependent component in the RSS values using data transformation. For instance, differential fingerprinting approaches rely on RSS differences, instead of absolute RSS values, to form the fingerprints [40,101,102]. These methods, however, suffer either from higher computational complexity, or poor localization performance, compared to traditional RSS fingerprinting methods. Thus, the challenge is to enable the use of *any* device during localization, while maintaining adequate accuracy.

Moreover, dealing with diverse devices is also very important in the increasingly popular crowdsourced fingerprinting systems. In general, crowdsourcing approaches rely on volunteers for collecting measurements on their smartphones in a participatory sensing fashion that they later contribute to the system [24]. This is also known as crowdsensing and the measurements may come from all sensors available on modern smartphones, including GPS trajectory data [147] or MAC addresses and RSS values from WLAN APs on a planet scale [65]. Specifically in crowdsourced fingerprinting systems, device heterogeneity comes naturally because users typically carry diverse mobile devices, including smartphones, PDAs, tablets, laptops, etc. Thus, cross-device measurements are incompatible, which renders the fusion of location-tagged RSS values from different devices in a single radiomap a great challenge.

1.5 Thesis Contributions

With respect to the research challenges identified in the field of localization and tracking in wireless networks, the contributions of this thesis are three-fold.

First, we address fault tolerant target localization and tracking in binary WSNs [76,83,84, 104,107]. We start by introducing a novel Markov Chain (MC) fault model that captures the spatiotemporal dynamics of sensor node faults and is capable of simulating various types of faults (e.g., temporary or permanent, reverse status and stuck at a particular value, spatially correlated faults) [83,84]. Moreover, we formulate the sensor health state estimation problem as a Hidden Markov Model (HMM) and devise efficient stochastic estimators to infer the unknown sensor states simultaneously with target tracking [76,84]. Then, we develop a closed loop architecture, referred to as ftTRACK, that combines sensor health state estimation, with target localization and location smoothing by means of Bayesian filtering. The ftTRACK architecture is a centralized solution that utilizes the spatiotemporal information provided by the sensor network to detect individual faulty sensors, which are subsequently excluded from the target tracking process [76]. Working towards a distributed ftTRACK architecture, we develop a fault tolerant approach that enables the identification of multiple targets and couple that with a distributed target localization algorithm [104,107].

Second, we deal with faults in WLAN fingerprinting localization systems [74,75,85,86].

Starting with fault modelling, we treat WLAN AP failures and attacks in a unified framework because they both inject faults that may lead to significant accuracy degradation during localization. In particular, we define realistic fault models that capture the effect of AP malfunctions or adversary attacks [74]. Regarding AP fault detection, we introduce RSS distance-based fault indicators [85], as well as likelihood-based indicators inspired by probabilistic fingerprinting methods, and evaluate their performance in terms of detection accuracy under various fault models [86]. Next, we develop a class of fault tolerant localization algorithms for WLAN fingerprinting systems. One approach is inspired by a fault tolerant method applicable in binary WSNs that we properly modify and adapt to the WLAN setup [75]. In another approach, we combine our fault detection mechanisms with modified distance and likelihood metrics to build hybrid fingerprinting algorithms that are robust to AP faults [86].

Third, we cope with device diversity in WLAN fingerprinting systems [87–89]. We present an innovative device self-calibration method that uses histograms of RSS values to fit a linear mapping between the reference device, which was used to create the RSS radiomap, and the heterogeneous user-carried device [87,88]. A linear relation between RSS values measured by heterogeneous devices has been reported in several experimental studies [52,62,87,88,116]. In addition, we propose a novel method based on RSS differences that performs considerably better than existing differential fingerprinting approaches, in terms of localization accuracy and computational complexity. Moreover, we formulate the problem of crowdsourcing in fingerprinting systems, introduce the notion of RSS differences in our formulation and evaluate various differential approaches to fuse heterogeneous RSS data into a single radiomap [89].

1.6 Thesis Structure

The remainder of this thesis is structured as follows.

In Chapter 2 we provide an overview of existing works related to localization and tracking in binary WSNs, fingerprinting localization in WLAN and localization for heterogeneous devices.

In Chapter 3 we deal with tracking in binary WSNs and present the ftTRACK architecture for robust target tracking in the presence of different types of sensor faults.

In Chapter 4 we focus on WLAN fingerprinting and describe our work on improving the resilience of existing algorithms to different types of network faults, starting with fault modelling and fault detection schemes and followed by fault tolerant fingerprinting algorithms.

In Chapter 5 we investigate device diversity in WLAN localization and discuss two different approaches, one that exploits RSS histograms and another one based of differential RSS fingerprints, for addressing this problem. Then, we extend our findings in crowdsourced systems.

In Chapter 6 we summarize the thesis contributions and provide concluding remarks, as well as directions for future work.

Chapter 2

Related Work

2.1 Localization and Tracking in Binary WSNs

2.1.1 Target Localization

Over the last 20 years there has been an increasing interest in event source or target localization in the context of WSNs. Several techniques have been proposed in the literature that exploit signal strength, angular or timing measurements and rely on arrays of sensors for radar, sonar and acoustic target localization applications; see [5, 27, 127] and references therein. These are also known as range-based techniques in which location is estimated in a least squares sense through multilateration, using a set of distances from at least three landmarks with known locations. The resistance of this class of techniques to distance spoofing attacks, e.g., by altering the RSS level that leads to erroneous distance calculation, is analysed in [19] and a mechanism for secure positioning, coined verifiable multilateration, is described.

In this thesis we focus on location estimators that use binary data, i.e., decisions made by binary sensors that simply compare their measurements with a predefined threshold. Among the various methods studied for localization in binary WSNs, the simplest is the Centroid Estimator (CE) [38], while another approach is based on the classical Maximum Likelihood (ML) estimator [113]. However, both methods do not consider sensor faults and may yield significant estimation errors when faults are present in the field. To this end, the Fault Tolerant Maximum Likelihood (FTML) estimator was recently proposed [103] and its performance is closely approximated by the Subtract on Negative Add on Positive (SNAP) algorithm, which has also low computational complexity [108]. Authors in [129] use geometric techniques that rely on the overlap of the sensing areas of sensors and employ a non-ideal sensing model for localization. On a different line, authors in [58, 59] achieve robust target localization by first using a Local Vote Decision Fusion (LVDF) scheme for correcting the original decisions based on the majority of the neighbouring sensors' decisions followed by numerical optimization techniques.

2.1.2 Target Tracking

Tracking techniques, as opposed to the localization techniques discussed previously, usually incorporate a target mobility model and assume a probability distribution for the sensor measurement errors to improve performance. They rely on Bayesian filtering variants [47], such as Kalman or particle filters, to mitigate the effect of measurement noise and alleviate high localization errors that do not reflect the target's mobility pattern. In general, tracking methods are classified into centralized, decentralized and distributed approaches.

In centralized methods, the sensor measurements are communicated to a central repository, where there is also a processing unit to run the localization and tracking algorithms [7, 39, 128, 129].

In decentralized approaches, a cluster is formed when a target is detected and all the sensors in the vicinity of the target forward their measurements to the cluster head (leader), which is responsible for estimating the target location [28, 60, 133].

In distributed approaches, each node exchanges messages only with its neighbours in the network and runs the tracking algorithm locally by using available measurements from all neighbouring nodes [26, 114].

2.1.3 Sensor Health State Estimation

Some related works address the sensor health state estimation problem [45, 111], however they assume that the raw signal generated by the source is sampled at the sensors, thus they cannot be applied directly in binary WSNs. Recently, trust index methods have been studied for detecting misbehaving sensors and reducing their impact on the underlying task [136]. For instance, authors in [141] reduce the effect of faulty or malicious sensors by decreasing their trust indices according to an exponential rule, while the target is moving inside the sensor field. However, trust index methods have some fundamental limitations. For example, once a sensor loses its trust it takes a long time to recover (i.e., regain its trust), while it might have been a temporary fault. Thus, it makes sense to re-evaluate the sensor state and make sudden decisions in real time. Importantly, sensors considered as faulty are still involved in the localization task, but with reduced weight, while a better approach would be to ignore them completely.

2.2 Fingerprinting Localization in WLAN

2.2.1 Standard Localization Methods

As already discussed, RSS is very convenient for inferring location in WLAN-based localization systems. In this case, a signal propagation model, such as the *log-distance* model [122], can be used to translate RSS values to distances from the respective WLAN APs and then the unknown location is determined through multilateration.

According to the simple *log-distance* propagation model, the RSS values in dBm are given by

$$RSS = K - 10f \log_{10} d + X, \tag{2.1}$$

where d denotes the distance between the transmitter (e.g., a WLAN AP) and the receiver (e.g., a mobile device), while the intercept term K provides the RSS value when $d = 1\,\text{m}$ and encapsulates device specific characteristics, such as the gain at the transmitter and the receiver antennas, the AP transmit power, etc. The coefficient f depends on the propagation environment, while $X \sim \mathcal{N}(0, \sigma^2)$ is the Gaussian noise disturbing the RSS values.

A major drawback of model-based WLAN localization is the requirement for exact AP locations, which are usually unknown in real-life applications. Moreover, the performance of propagation models in indoor environments is questionable. This is primarily due to NLOS conditions and the existence of multipath components. Several methods employ propagation models that are calibrated to best fit the specific environment in the target area. These methods try to express the effects of walls, ceilings or people moving, through a set of model parameters, which are estimated using a large amount of RSS data collected by mobile devices throughout the area [9, 33, 67, 130]. Similarly, recent methods calibrate the propagation model by using measurements collected at a few known user locations [36]. Alternatively, the model parameters can be estimated by measuring the device's signal strength received at the surrounding APs [51], or by taking pairwise measurements between the deployed APs [10] or by using RSS measurements among APs and user devices [96]. However, these model calibration methods require firmware modifications at the APs. Thus, it is difficult to apply such methods on off-the-shelf APs.

Even though indoor propagation models can be calibrated to some extent, it is extremely hard to model accurately other factors that affect the signal strength level, such as the user orientation or the use of directional antennas in the WLAN APs. To this end, fingerprinting localization methods address the limitations of model-based approaches by exploiting a discrete RSS profile of the target area. This comes at the expense of data collection time and effort. Essentially, the fingerprints contain RSS values from neighbouring WLAN APs and are associated with a set of predefined reference locations. The unknown user location can then be estimated by finding the best match between the observed fingerprint during localization and the reference fingerprints. An overview and taxonomy of RSS fingerprinting methods is available in [63].

A wide variety of fingerprinting methods have been presented so far. Deterministic methods, including the Nearest Neighbour (NN) approach [9, 10], estimate user location as a convex combination of the reference locations [55, 94, 120, 144]. In probabilistic approaches the user location is treated as a random vector that can be estimated by calculating the conditional probabilities (posterior) of being at a particular location given the observed RSS fingerprint during localization [22, 52, 70, 71, 100, 115, 125, 146].

The fingerprinting research community has also employed computational intelligence methods, including fuzzy logic [1, 2, 4, 8, 48, 49, 134, 135]. For instance, a two-stage fuzzy logic method first investigates a number of candidates of proximate calibration points (i.e., reference locations) and subsequently the second stage is responsible for an appropriate weighting of their coordinates to yield the positioning result [134, 135]. Authors in [48, 49] present a fuzzy logic-based system which employs an incremental lifelong learning approach for adjusting its behaviour to the varying and changing WLAN signals in order to localize a given user in ambient intelligent environments. The approach presented in [1, 2] extends the *Fuzzy ArtMap* neural network system [20] to enable on the fly expansion and reconstruction of location systems. A fuzzy location algorithm using fuzzy inference systems with WLAN RSS measurements is presented in [8], while tracking is performed by means of a fuzzy automaton. Authors in [4] describe a hybrid localization solution that combines a multi-variable fuzzy inference system with a multi-NN algorithm.

Other approaches employ decision trees [32, 145] or artificial neural networks, including Multi Layer Perceptron (MLP) designs [12, 16, 17, 43, 126], Radial Basis Function Networks (RBFN) [73, 77, 81, 91] and Generalized Regression Neural Networks (GRNN) [110].

All these fingerprinting methods, however, do not consider AP faults during localization.

2.2.2 Fault and Attack Models

Some early works investigate the performance of localization algorithms when a single AP is shut down, either intentionally or accidentally. For instance, authors in [120] evaluate several variants of the NN method and weighting schemes in a multi-floor area covered by 10 APs to determine their robustness when the AP, which is closest to the mobile device during localization, becomes unavailable. In [54] the effect of eliminating one out of five APs in the position estimation accuracy is studied using Monte Carlo simulations based on IEEE 802.11 channel models. Both works reach the conclusion that NN approaches, especially if more than one neighbours are used, are quite robust to single AP failures.

In [95] a signal attenuation or amplification attack is simulated by randomly choosing the RSS readings of one or two out of six APs and multiplying them with a constant. Authors in [30] consider a similar linear attack model which is simulated by perturbing the original RSS values over all APs by a constant value. It was observed that using a real material, such as glass, metal, foil, books, etc., causes a constant percentage power loss that is independent of distance. This type of attacks is easy to launch with low cost materials and at the same time the adversary may control the effect of the attack by selecting the appropriate material [31]. On the other hand, amplification attacks can be performed by deliberately increasing the AP transmit power. Another attack model assumes that RSS measurements are corrupted by additive Gaussian noise with higher variance [69]. Under this model, an RSS attack is caused by altering the propagation environment and is simulated by adding more noise to the collected test data.

2.2.3 Fault and Attack Detection

Fault tolerant localization systems could be supported by fault (attack) detection mechanisms that are efficient, i.e., exhibit high detection and low false positive rates. For instance, a detection component could trigger an alert for the security personnel each time there is a fault (attack) indication. Furthermore, the positioning component could also switch to a fault tolerant counterpart for mitigating the effect of the fault (attack) and still provide adequate level of accuracy until the problem is resolved.

Attack detection in wireless localization is studied in [29, 31] for a variety of localization methods, including range-based and RSS fingerprinting, and detection relies on statistical significance testing. For example, in the case of the NN fingerprinting method, the minimum distance between the observed fingerprint during localization and the fingerprints in the pre-constructed radiomap, denoted as D_s, is used as the test statistic. The distribution of the training data contained in the radiomap is used to select an appropriate threshold τ and subsequently an attack is signified during localization in case $D_s > \tau$. A key observation in this work is that the performance of the proposed detection method is better under signal amplification attacks, compared to signal attenuation attacks. For probabilistic fingerprinting techniques equivalent test statistics are studied, including the likelihood of the location with the highest value or the sum of the likelihoods over all locations. Both test statistics are found to decrease significantly under attack.

Authors in [97] exploit the communication capabilities among transmitters in a WSN setup based on MicaZ beacon nodes to decide whether there are node failures in the system. In this approach, beacon nodes periodically measure their local neighbourhood, defined as the set of other beacon nodes that they can communicate with. This neighbourhood is compared to the original neighbourhood, which is measured shortly after the system has been installed. If the intersection between the current and original neighbourhoods is large, the system is assumed to be fault-free. On the other hand, if the fraction of failed nodes exceeds some threshold, then failure (or similarly an attack) is detected. However, this approach assumes adequate connectivity between beacon nodes that does not change substantially over time. Moreover, due to the node communication requirement, this approach cannot be directly applied to localization methods that rely on WLAN APs. Recently, we have proposed to leverage the inertial sensors found in modern smartphones (i.e., accelerometer, gyroscope, and digital compass) to track the user in the background by means of Pedestrian Dead Reckoning (PDR) and use the PDR location stream for detecting AP failures in the fingerprinting system [53].

2.2.4 Fault Tolerant Localization Methods

Traditional fingerprinting methods do not consider faults. Some early works examine performance when a single AP is shut down or eliminated [54, 120], while it is reported that the accuracy can be severely degraded when the percentage of faulty APs in the system is increased [85]. Thus, it is essential to build fault tolerant algorithms to guarantee that the localization error does not increase rapidly in the presence of faults.

As a first step to improve the robustness of the localization system to RSS-based attacks, authors in [95] suggest to increase redundancy by using more sensors or APs. Moreover, the effect of outlier APs is reduced with the introduction of a median-based, instead of the Euclidean, distance metric that is applicable in both range-based and fingerprinting localization methods. Similarly, in the context of the *MoteTrack* system [97], the Euclidean distance in the NN algorithm is replaced by an adaptive fingerprint distance metric to cater for faulty nodes. Under the presence of faults, the adaptive metric penalizes only RSS values found in the currently observed fingerprint and not in a reference fingerprint, so as to minimize the errors introduced from failed nodes[1]. In the fault-free case, the algorithm reverts to the standard metric, thus penalizing RSS values from all nodes not found in common between the observed and reference fingerprints. On a different line, Kushki et al. [69] describe a sensor selection methodology, based on a nonparametric estimate of the Fisher Information, for increasing the resilience of fingerprinting systems to RSS attacks. Essentially, this method selects only a number of reliable APs from the set of available APs to mitigate the attack.

2.3 Cross Device Localization

Device independent localization has recently attracted researchers' interest, due to the requirement for the provision of accurate and reliable location estimates regardless of the device carried by the user. Several works assume that the fingerprinting radiomap has been created using RSS data collected with a single device and try to address the device diversity issue, that arises when the user performs a location request with a different device, mainly through signal strength data fitting approaches or calibration-free techniques.

Moreover, the device heterogeneity issue comes naturally in crowdsourced fingerprinting systems, where the radiomap creation is facilitated through user collaboration [15, 72, 117]. This is because volunteers typically carry diverse mobile devices that do not report the RSS values in a similar way. Thus, cross-device measurements are incompatible. Crowdsourced fingerprinting systems try to deal with heterogeneous devices to enable the fusion of location-tagged RSS values from different devices in a single usable radiomap.

2.3.1 Signal Strength Data Fitting Methods

The RSS data fitting methods try to create a mapping between different devices and are motivated by the linear relation between the RSS values reported by heterogeneous devices, which has been observed experimentally in several studies [52, 62, 90].

[1]The intuition behind this statement will become clear later during the description of our fault tolerant localization algorithms in Section 4.4.

One approach, referred to as *manual calibration*, is to collect a series of RSS measurements at several *known* locations with a pair of devices and subsequently estimate the linear fitting parameters. Essentially, if a sufficient number of colocated RSS pairs (i.e., collected at the same location and time with two different devices) is available, then the linear parameters can be estimated through standard least squares fitting [52, 62, 90]. Authors in [46] report that the relationship between the RSS values of different devices can sometimes be nonlinear and use a Support Vector Regressor (SVR) and a GRNN as learning algorithms. Nevertheless, the *manual calibration* method requires a considerable data collection effort by the user prior to positioning. To reduce this effort, authors in [143] investigate the case where a very small labelled calibration dataset is collected with the user-carried device and exploit linear adaptation on Gaussian processes. When few location-tagged samples are available for each different device, a multi-task SVR algorithm is used to learn a device-specific mapping in a common latent, i.e., lower dimension, subspace [149]. However, it is difficult for the SVR to deal with missing RSS values, e.g., due to partial AP coverage or transient effects [143]. Similarly, a dimensionality reduction method is described in [131], which learns a mapping between a source dataset and a target dataset in a low-dimensional space, so that the knowledge can be transferred between devices using the mapping relationship. Recently, we proposed the use of RSS data generated by a 3D Ray-Tracing simulation tool to build the fingerprinting radiomap and investigated the localization accuracy achieved with the *manual calibration* method, when these artificial data are used with heterogeneous devices [123, 124].

Even though the time and labour overhead for calibrating a new device can be significantly reduced when the localization area is covered by several APs [87], *manual calibration* still has limited applicability in real-life applications. In a typical scenario, where users enter an indoor environment, such as shopping malls, airports, etc., carrying an uncalibrated device, they have to be guided to specific *known* locations for collecting RSS data. This implies that the users are already familiar with the area, which is usually not the case.

Device calibration with RSS data recorded by the user at *unknown* locations is feasible, but computationally expensive methods are required to obtain the linear fitting parameters. For instance, the parameters can be estimated by maximizing the confidence value produced by Markov localization [52] or through a weighted least squares method [62]. These are known as *quasi-automatic calibration* methods.

In *automatic calibration* methods, RSS data collected at *unknown* locations are used as in the *quasi-automatic calibration* case. The objective, however, is to minimize the user intervention and ideally perform localization and device calibration simultaneously, while the user walks freely inside the area of interest. To this end, an Expectation-Maximization (EM) algorithm is proposed in [52]. Alternatively, authors in [62] detect when the user is stationary during localization in order to divide the data into parts which come from the same *unknown* location and then use these data with a *quasi-automatic calibration* approach. An unsupervised learning method uses the Pearson product-moment correlation coefficient to label the RSS readings with a rough location estimate, while the user is walking, and then employs EM and neural network learning algorithms to obtain the linear fitting parameters [138]. Authors in [61] also assume a linear fitting function and localization is performed when a signal peak is detected, while the user location is estimated as the location of the maximum RSS recorded during the training phase. This approach is justified by the fact that the location of the maximum RSS is preserved, even though the RSS range varies significantly among heterogeneous devices. A major limitation, however, is that the location can be determined only when a signal peak is detected. Authors in [13] assume that the RSS values from heterogeneous devices differ only by a common factor (offset) and incorporate the online estimation of this factor in the likelihood function of their probabilistic localization algorithm.

The possibility of using RSS histograms for *automatic calibration* is mentioned without implementation details in [66]. Authors in [109] create the empirical Cumulative Distribution Function (CDF) for several devices using the RSS values collected at *known* locations and then use the inverse CDF function, instead of least squares fitting, to build a database of device models that map the RSS values of the user device to the reference device. However, this method cannot easily scale to a large number of device pairs, while the selection of the appropriate model during localization is based on the existence of an easily distinguishable location (e.g., building entrance or exit) that may never be visited by the user.

2.3.2 Calibration-free Methods

The approaches that fall under this category try to remove the device-dependent component in the RSS values through data transformation. For instance, differences between RSS values can be used to form the fingerprints, instead of absolute RSS values. This effectively removes the constant term K in the *log-distance* model (2.1) and makes RSS differences from diverse devices compatible with each other. The differential fingerprints can be created by taking the difference between all possible AP pairs, i.e., if there are n APs inside the area of interest then the transformed fingerprint contains $\binom{n}{2}$ RSS differences [40, 102]. However, this method may increase dramatically the dimensionality of the fingerprints, especially is areas covered by a large number of APs, thus leading to higher computational complexity compared to the traditional RSS fingerprints. To address this issue, the Signal Strength Difference (SSD) approach creates the fingerprints by subtracting the RSS value of an anchor AP from the other RSS values [101], so that the differential fingerprints contains only the $n - 1$ RSS differences that are independent. The anchor AP can be selected as the one that exhibits the least average deviation of RSS values over the whole localization area [56]. However, selecting the anchor AP is not trivial, especially in large scale setups where the APs do not provide ubiquitous coverage. In general, RSS differences exhibit higher noise variance, compared to traditional RSS values, which may degrade the localization accuracy, especially if the localization device is the same with device used to build the radiomap [87, 101].

The Hyperbolic Location Fingerprinting (HLF) approach combines normalized logarithm ratios of the RSS power from different APs to remove the device-dependent component [64]. However, the resulting RSS logarithm ratios are not totally free from the intercept term K in the propagation model (2.1), thus they cannot fully mitigate the hardware variations [101].

On a different line, RSS ranking methods rely on ranked, rather than absolute, RSS values (i.e., RSS values from a set of APs are ranked from high to low). The intuition is that the ranking of RSS values is not affected by device-specific hardware features [35, 99]. However, Ranked-Based Fingerprinting (RBF) is expected to perform worse, compared to standard RSS fingerprinting, because the fine-grain information of the RSS levels is lost when ranks are used as indicated by our experimental evaluations [88].

2.3.3 Crowdsourced Localization Systems

Crowdsourcing has recently emerged as a viable solution to address the maintenance cost, as well as scalability issues, related to the RSS radiomap. Some early systems employed the idea of crowdsourcing to expand a core radiomap created by trained contributors. For instance, the *Active Campus* project developed a user-assisted system that employs user feedback for fast, accurate and low-maintenance localization [14]. Along the same line, the system presented in [23] reduces the radiomap creation effort by merging user-supplied data with an initial radiomap set up by the system operator.

On the other hand, *Place Lab* is a fully crowdsourced solution, i.e., it does not require an initial radiomap, although it does not rely on RSS fingerprinting, but rather uses a Google Maps wardriving approach for populating a database with approximate AP coordinates [72]. *Redpin* is one of the first attempts to build a fingerprinting system that relies entirely on user collaboration [15]. Other fully crowdsourced localization systems have also been presented in [11, 50, 93], however, all these systems do not consider device heterogeneity.

The *WiFiSLAM* application, which was recently acquired by Apple [41], allows the user to carry any device during localization. Still, a homogeneous radiomap, i.e., built from RSS data collected with a single device or by several contributors carrying the same device, is required. Similarly, the *Molé* system relies on a homogeneous radiomap and applies a linear transformation on the signal strengths values, followed by kernelisation of the RSS histogram to support different user devices during localization [90]. The *Zee* system enables crowdsourcing of location-annotated WLAN measurements, without requiring any active user intervention in terms of location input or placement of the phone [121]. Yet, these systems cannot exploit the full benefit of crowdsourcing, as they do not address the radiomap creation using diverse devices.

The *Elekspot* system deals with device heterogeneity when the radiomap is built and uses linear relations among device pairs based on duplicated contributions in the same locations, while the linear parameters are maintained in a square matrix for all devices [92]. However, this approach relies on the condition that enough duplicated contributions are made. Alternatively, authors in [34] use standard clustering algorithms to put *similar* devices in the same cluster, so that they can share the fingerprints among them. In case a new device wants to contribute data to the system, they employ an EM algorithm to learn the linear fitting parameters for matching the best cluster. Finally, the *FreeLoc* system handles heterogeneous data by using relative, rather than absolute, RSS values in the radiomap fingerprints [142]. This approach is similar to ranked RSS values, thus the fine-grain information of the RSS values is lost and the quality of the crowdsourced radiomap may deteriorate.

2.4 Chapter Summary

In this chapter we provide a literature review and outline the research that is related to our work.

Firstly, we overview target localization and tracking algorithms in binary WSNs and discuss existing sensor health state estimation approaches.

Subsequently, we put our focus on WLAN fingerprinting localization and start with a short survey of standard algorithms that do not consider faults. In the following, we present relevant works that aim to improve the resilience of fingerprinting algorithms to network faults or attacks through fault and attack modelling, fault and attack detection, as well as fault tolerant localization.

Finally, we discuss methods addressing the device heterogeneity problem that is inherent in WLAN-based localization systems, especially those systems leveraging crowdsourced data collected with a variety of user-carried devices.

Chapter 3

Fault Tolerant Tracking in Binary WSNs

3.1 Background

3.1.1 Sensor Network Model

We make the following assumptions, which are common and reasonable for monitoring applications using WSNs:

1. A set of N static sensor nodes is uniformly spread over a rectangular field $\mathcal{A}$ and their location $\ell_n = (x_n, y_n)$, $n = 1, \ldots, N$ is assumed to be known.

2. A single event source[1] is moving at steady speed inside $\mathcal{A}$ and at time t it is located at $\ell_s(t) = (x_s(t), y_s(t))$.

The alarm status of the n-th sensor node $n = 1, \ldots, N$ at every time step t, $t = 1, \ldots, M$, is given by

$$A_n(t) = \begin{cases} 0 & \text{if } z_n(t) < T \\ 1 & \text{if } z_n(t) \geq T \end{cases}, \tag{3.1}$$

where $z_n(t)$ is the signal present at sensor n and T is a predefined threshold. If sensor node n is alarmed, i.e. $A_n(t) = 1$, it sends a packet to the sink, otherwise it remains silent. In this thesis, we assume that the target emits a constant signal c which is attenuated inversely proportional to the distance from the target raised to power $\zeta \in \mathbb{R}^+$. The signal $z_n(t)$ is given by

$$z_n(t) = \frac{c}{1 + d_n(t)^{\zeta}} + w_n(t), \tag{3.2}$$

where $w_n(t)$ is additive white Gaussian noise, i.e., $w_n(t) \sim \mathcal{N}(0, \sigma_w^2)$ and $d_n(t)$ is the radial distance between sensor node n and the target at time t, i.e., $d_n(t) = \sqrt{(x_n - x_s(t))^2 + (y_n - y_s(t))^2}$.

[1]The terms *source* and *target* are used interchangeably.

The parameter ζ depends on the environmental conditions, while the threshold T is chosen large enough to minimize the probability of the sensor being falsely alarmed, e.g., as a result of noise. The probability of false alarms P_{fa} is the probability that at least one of the sensors mistakenly reports the presence of a source and is given by

$$P_{fa} = 1 - \prod_{i=1}^{N} \Phi(\frac{T}{\sigma_w}), \tag{3.3}$$

where $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp(-\frac{y^2}{2}) dy$ is the CDF of the standard Gaussian random variable $\mathcal{N}(0,1)$. Moreover, the optimal threshold can be selected to minimize the following cost function J representing the overall error in detection as a function of the threshold T

$$J(T) = wP_{fa} + (1-w)P_{nd}, \tag{3.4}$$

where P_{nd} is the probability of no detection and $0 \leq w \leq 1$ is a user specified weight that should be chosen with care according to the application [105]. A small w implies that the application can tolerate more false alarms, but it cannot tolerate any missed events. For example, in networks that monitor for toxic terrorist attacks in crowded areas, w could be set close to zero. On the other hand, larger values of w imply that some missed events may be tolerated to reduce the cost of false alarms. For example, in applications such as environmental monitoring of large areas, false alarms may incur a significant cost because a response crew may have to travel to the suspected area. Also, in many cases, frequent false alarms may make the users simply ignore all alarms and as a result important events may go undetected. Alternatively, one could also formulate the problem as a constrained optimization problem, i.e., minimize P_{fa} subject to $P_{nd} < P_{nd}^{des}$ for some desired value P_{nd}^{des}. More details about the computation of the threshold can be found in [105].

Next, we define the footprint of the events.

Definition 1. *The Region of Influence (ROI) of a target is the area around the target location inside which a sensor node will be alarmed with probability at least 0.5*[2].

For the model of (3.2) the *ROI* becomes a disc centred at the target location with radius $R_I = \sqrt[\zeta]{c/T - 1}$.

[2]Using 0.5 is a convenient way to define the source *ROI* because the *ROI* size becomes independent of the noise variance when dealing with noise that has symmetric distribution function, e.g., Gaussian, [108].

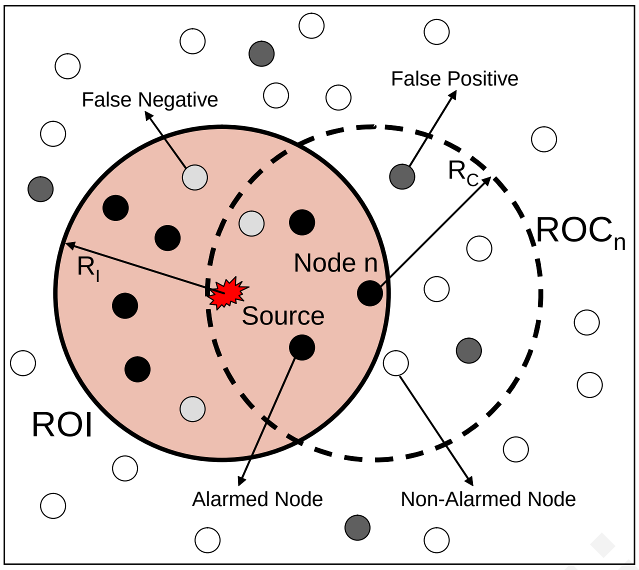


Figure 3.1: Different regions with respect to the sensor and target locations and behaviour of faulty sensors in binary WSNs.

Definition 2. *The Region of Coverage of sensor node n (ROC_n) is the area around the location of sensor node n inside which if a target is present, then it will be detected with high probability (at least 0.5).*

By symmetry, for the single source case, ROC_n is a disc centred at the alarmed sensor node location with radius R_C, while $R_C \equiv R_I$, $\forall n$. The different regions with respect to the sensor and target locations in our WSN model are illustrated in Figure 3.1.

3.1.2 Sensor Node Fault Types

In applications with dense node deployments over large areas it is expected that at any point in time, a set of sensors may not be functioning correctly due to some fault. A fault model commonly used by the research community assumes that each node can exhibit erroneous behaviour with some probability P_f, i.e., the sensor state is temporally independent, and in this case its original observation is reversed [98,106,108]. However, this model may not fully capture the non-zero autocorrelation of the stochastic process that drives the health state of the sensor because in real WSNs applications the nodes may spend several time steps being healthy before becoming faulty and vice versa. For instance, the measurements provided by a sensor may get stuck at a high or low value for a period of time, e.g., due to board overheating or battery depletion [6, 112], that manifest themselves as either temporary or permanent faults. An example of a temporary power fault is when the batteries of a node connected to a photovoltaic deplete after prolonged cloudy periods and are then recharged when the sun comes up. If no photovoltaic is available to recharge the node, then battery depletion results in a permanent fault.

For the purposes of this thesis we assume that a sensor may suffer from the following three types of faults.

Reverse Status (RS) faults: Binary sensors may report the opposite readings than the correct ones due to software bugs because such nodes are often reprogrammed wirelessly. Alternatively, in a sophisticated attack scenario, the attacker may have compromised a number of sensors or deployed a malicious sensor network, so that several sensors deliberately reverse their original output to prevent intruder detection [37]. In this case, faulty sensors that fall inside the *ROI* of the target become non-alarmed; these are denoted as *false negatives* and are shown as light gray circles in Figure 3.1. On the other hand, sensors located outside the *ROI* become alarmed and are denoted as *false positives* (dark gray circles).

Stuck-At-1 (SA1) faults: A sensor may exhibit a stuck-at fault, i.e., the sensor measurements get stuck at a specific value for a period of time. This fault can be attributed to processing board overheating [6] or low battery level [112]. For binary sensors, a node may constantly report the presence of a target during a SA1 fault, i.e., a series of *false positives*, in case the stuck-at value exceeds the detection threshold T. Similar behaviour is observed if the threshold T is wrongly programmed, e.g., the threshold value is set below the noise level. Moreover, a possible attacker strategy could be the deployment of small decoy sources to make the actual intruder detection more difficult [132].

Stuck-At-0 (SA0) faults: These faults reveal themselves when sensor node n fails to report the presence of a target inside its ROC_n, i.e., a series of *false negatives*. They occur when the sensor measurements are stuck at a value below the detection threshold T, e.g., due to power depletion. They also appear in case of dropped packets, because packets from an alarmed sensor that do not reach the sink are equivalent to the target not being detected by that particular sensor. SA0 faults may also appear by unintentionally setting the detection threshold at a very high value, so that it is never exceeded even if the target is in close proximity to the sensor. Another scenario captured by our SA0 fault model is the offset bias, which is a very common sensor error [68]. Offset bias alters the sensor measurements uniformly by a certain value. For example, the error could result that a light sensor always reads only 60% of the correct luminance due to dust or other obstacles accumulated on a subset of its sensor cells. In such a case a binary light sensor would be falsely non-alarmed (i.e., suffers a SA0 fault) while a light source approaches.

3.1.3 Sensor Node Fault Model

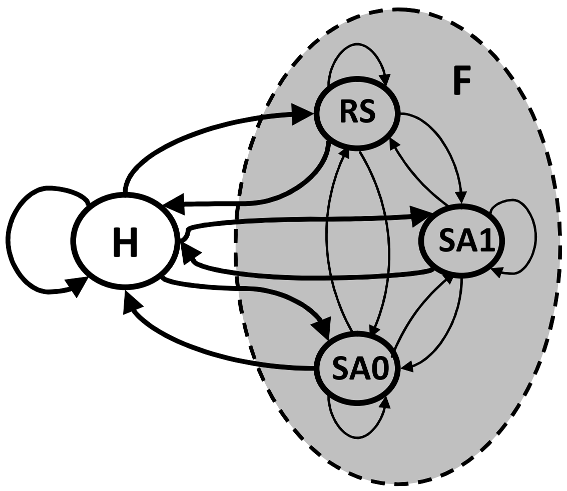
We capture the temporal state of each sensor by assuming that its health state is represented with a MC. In our preliminary works we assumed that each sensor can be either at *healthy* or *faulty* state, i.e., a simple two-state MC model, [83, 84]. To take into account the different fault types described previously, we consider a four-state MC model that features three distinct faulty states, as shown in Figure 3.2a. In this case, the state probabilities vector of sensor n at time step t is $\pi_n(t) = [\pi_n^{RS}(t)\ \pi_n^{SA1}(t)\ \pi_n^{SA0}(t)\ \pi_n^{H}(t)]^T$, i.e., $\pi_n^i(t) = \mathbf{P}[s_n(t) = i]$, $i \in \{RS, SA1, SA0, H\}$ where $\pi_n^{RS}(t) + \pi_n^{SA1}(t) + \pi_n^{SA0}(t) + \pi_n^{H}(t) = 1$ and $(\cdot)^T$ indicates the transpose. The state transition matrix is

$$C = \begin{bmatrix} p^{RS,RS} & p^{RS,SA1} & p^{RS,SA0} & p^{RS,H} \\ p^{SA1,RS} & p^{SA1,SA1} & p^{SA1,SA0} & p^{SA1,H} \\ p^{SA0,RS} & p^{SA0,SA1} & p^{SA0,SA0} & p^{SA0,H} \\ p^{H,RS} & p^{H,SA1} & p^{H,SA0} & p^{H,H} \end{bmatrix},$$

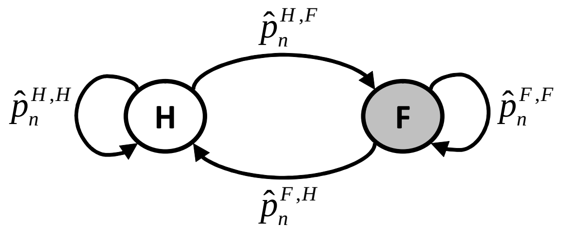
where $p^{i,j},\ i, j \in \{RS, SA1, SA0, H\}$ is the transition probability from state i to state j. In this model, the MC dynamics are given by $\pi_n(t+1) = C^T \pi_n(t)$, while the steady state probabilities are denoted as $\pi_n^i = \lim_{t\to\infty} \mathbf{P}[s_n(t) = i]$, $i \in \{RS, SA1, SA0, H\}$. When sensor n is in the healthy state, its alarm status $A_n(t)$ is given by equation (3.1), otherwise when the sensor is in one of the faulty states its alarm status is determined based on the corresponding *type* of fault.

We now demonstrate the capability of our fault model to generate various real faults, either temporary or permanent, that have been documented in the literature [112]. For instance, assuming that there are no transitions between faulty states and setting the remaining transition probabilities as $p^{H,H} = 0.91$, $p^{RS,H} = 0.2$, $p^{SA1,H} = 0.4$, $p^{SA0,H} = 0.4$ and $p^{H,j} = 0.03,\ j \in \{RS, SA1, SA0\}$, we can simulate different types of failures that appear simultaneously and are also non uniform, e.g., a sensor that suffers from a RS fault is less likely to recover, compared to SA1 and SA0 faults. In this case, $[\pi_n^{RS}\ \pi_n^{SA1}\ \pi_n^{SA0}\ \pi_n^{H}]^T = [0.11\ 0.06\ 0.06\ 0.77]^T$ indicating that each sensor is expected to spend 77% of time behaving normally and 23% of time being faulty, with more time spent at the RS state.

In general, one can generate any desired scenario by properly adjusting the transition probabilities in matrix C and solving the steady state equation of the MC. Moreover, permanent faults can be generated by making the faulty states absorbing, i.e., $p^{i,i} = 1,\ i \in \{RS, SA1, SA0\}$. In addition, by adjusting the probability $p^{H,H}$ we can control how much time the sensor behaves normally before it exhibits the permanent fault. Finally, we point out that it is easy to apply various fault models at the same time, each one with different transition



(a) Fault model with multiple faulty states.



(b) State estimation model with grouped faulty states.

Figure 3.2: Markov Chain sensor node fault and health state estimation models.

probabilities, in order to control the behaviour of individual sensors or clusters of sensors. In this way, one can generate spatially correlated faults by using the same MC for a subset of neighbouring sensors.

3.1.4 Target Mobility and Measurement Model

We assume that the target is traversing the sensor field according to a linear discrete-time process disturbed by noise

$$X(t) = \underbrace{\begin{bmatrix} I_2 & T_s I_2 \\ 0 & I_2 \end{bmatrix}}_{\Phi} X(t-1) + \underbrace{\begin{bmatrix} \frac{T_s^2}{2} I_2 \\ T_s I_2 \end{bmatrix}}_{\Gamma} W(t-1) \tag{3.5a}$$

$$Y(t) = \begin{bmatrix} I_2 & 0 \end{bmatrix} X(t) + U(t), \tag{3.5b}$$

where $X(t) = \left[x_s(t)\ y_s(t)\ u_x(t)\ u_y(t)\right]^T$ represents the process state at time step t, i.e., the combined vector of the target position $[x_s(t)\ y_s(t)]^T$ and velocity $\left[u_x(t)\ u_y(t)\right]^T$. The measurement vector $Y(t)$ in equation (3.5b) only measures the target location, which is estimated by the underlying target localization algorithm, while $W(t)$ and $U(t)$ represent the process and measurement Gaussian noise, respectively. The sampling interval is denoted by T_s and I_2 is the 2×2 identity matrix.

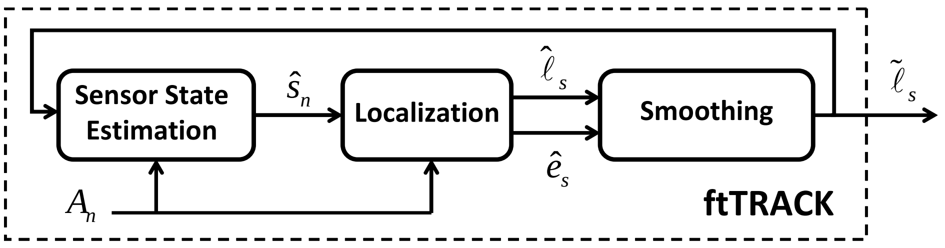


Figure 3.3: Block diagram of the ftTRACK target tracking architecture.

3.2 The ftTRACK Architecture

The proposed fault tolerant target tracking architecture (ftTRACK) consists of three main components connected in a closed loop, as indicated in Figure 3.3 [76]. Given the latest target location estimate $\tilde{\ell}_s(t) = (\tilde{x}_s(t), \tilde{y}_s(t))$, the *Sensor State Estimation* component determines the health state of each sensor $\hat{s}_n(t),\ i = 1, \ldots, N$ as either healthy or faulty. Subsequently, the sensor states are passed to the *Localization* component which uses information only from those sensors estimated as healthy in order to compute the current target location $\hat{\ell}_s(t) = (\hat{x}_s(t), \hat{y}_s(t))$ and the uncertainty $\hat{e}_s(t)$, i.e., localization error, associated with this location estimate. Finally, the *Smoothing* component filters the current location estimate, by means of a particle filter, to obtain a less noisy target location estimate. Next, we present the three components of the ftTRACK architecture in more detail.

We point out that we assume a centralized network architecture, where ftTRACK runs on the sink that collects all sensor observations at every time step. To address scalability issues in terms of increasing number of moving sources and varying sensor node density, in Section 3.4 we extend this approach to a distributed architecture, where ftTRACK would run locally on the current leader node to process the series of target location estimates. When the target is about to leave the range of the current leader, a new tracking region is created with the election of a new leader and the former leader propagates to the next one all the sensor health state information and filter parameters required to continue the ftTRACK computations.

3.2.1 Sensor Health State Estimation

In this section, we present a Bayesian approach for estimating the sensor health state $\hat{s}_n,\ n = 1, \ldots, N$. The estimator can be in one of two states (*healthy* or *faulty*), as shown in Figure 3.2b. It is assumed that the transitions from one state to the other are based on a

HMM with an estimated transition matrix

$$\hat{C}_n(t) = \begin{bmatrix} \hat{p}_n^{F,F}(t) & \hat{p}_n^{F,H}(t) \\ \hat{p}_n^{H,F}(t) & \hat{p}_n^{H,H}(t) \end{bmatrix}, \tag{3.6}$$

where $\hat{p}_n^{i,j}(t),\ i,j \in \{F,H\}$ is the estimated probability that the sensor health state estimator will transition from state i to state j. The intuition for using a two-state MC is that we are only interested to estimate whether sensor n is faulty in order to exclude it from the tracking task. Identifying the exact fault type is beyond the scope of this thesis and we leave it as future work.

We also denote the estimated sensor state probabilities at time step t as $\hat{\pi}_n(t) = [\hat{\pi}_n^F(t)\, \hat{\pi}_n^H(t)]^T$, i.e., $\hat{\pi}_n^i(t) = \mathbf{P}[\hat{s}_n(t) = i],\ i \in \{F,H\}$. The observations used by the estimator to update its state are given by the error signal $r_n(t)$.

Definition 3. *The error signal $r_n(t) \in \{0,1\}$ is computed based on the sensor alarm status and the distance from the actual target location. This error signal is formally given by*

$$r_n(t) = \begin{cases} 1 & \textit{if } d_n(t) \le R_I \textit{ AND } A_n(t) = 0 \\ 1 & \textit{if } d_n(t) > R_I \textit{ AND } A_n(t) = 1 \\ 0 & \textit{if } d_n(t) \le R_I \textit{ AND } A_n(t) = 1 \\ 0 & \textit{if } d_n(t) > R_I \textit{ AND } A_n(t) = 0 \end{cases}, \tag{3.7}$$

where $d_n(t) = \|\ell_n - \ell_s(t)\|$ denotes the distance between the location of the n-th sensor ℓ_n and the actual target location $\ell_s(t)$.

In particular, the sensor output at time t is *wrong* ($r_n(t) = 1$), in case sensor n is inside the circular *ROI* with radius R_I centred at $\ell_s(t)$ and is non-alarmed or in case the sensor is outside the *ROI* and is alarmed. On the other hand, the sensor output is *correct* ($r_n(t) = 0$), in case sensor n is inside the *ROI* and is alarmed or in case the sensor is outside the *ROI* and is non-alarmed.

We obtain $\hat{s}_n(t+1)$ by calculating the probability of a sensor being at a specific state given the current error signal, i.e., $\hat{\pi}_n^{i|q}(t) = \mathbf{P}[s_n(t) = i | r_n(t) = q],\ i \in \{F,H\},\ q \in \{0,1\}$. In this case, the maximum likelihood sensor state estimate is

$$\hat{s}_n(t+1)|_{r_n(t)=q} = \arg\max_{i \in \{F,H\}} \hat{\pi}_n^{i|q}(t),\ q \in \{0,1\}. \tag{3.8}$$

Using Bayes' rule, we compute $\hat{\pi}_n^{i|q}(t)$ as

$$\hat{\pi}_n^{i|q}(t) = \frac{\mathbf{P}[r_n(t) = q | s_n(t) = i]\hat{\pi}_n^i(t)}{\mathbf{P}[r_n(t) = q]} \tag{3.9a}$$

$$\hat{\pi}_n^{i|q}(t) = \frac{\mathbf{P}[r_n(t) = q | s_n(t) = i]\hat{\pi}_n^i(t)}{\sum_{j \in \{F,H\}} \mathbf{P}[r_n(t) = q | s_n(t) = j]\hat{\pi}_n^j(t)}. \tag{3.9b}$$

For convenience, we define the following two probabilities of a sensor having a *wrong* output.

Definition 4. *The probability of sensor n having wrong output at time t given that its state is Healthy is* $p_n^h(t) = \mathbf{P}[r_n(t) = 1 | s_n(t) = H]$.

The probability of sensor n having wrong output at time t given that its state is Faulty is $p_n^f(t) = \mathbf{P}[r_n(t) = 1 | s_n(t) = F]$.

Using these definitions in equation (3.9b), we get the following conditional probabilities for sensor n

$$\hat{\pi}_n^{F|1}(t) = \frac{p_n^f(t) \cdot \hat{\pi}_n^F(t)}{p_n^f(t) \cdot \hat{\pi}_n^F(t) + p_n^h(t) \cdot \hat{\pi}_n^H(t)} \tag{3.10a}$$

$$\hat{\pi}_n^{H|1}(t) = \frac{p_n^h(t) \cdot \hat{\pi}_n^H(t)}{p_n^f(t) \cdot \hat{\pi}_n^F(t) + p_n^h(t) \cdot \hat{\pi}_n^H(t)} \tag{3.10b}$$

$$\hat{\pi}_n^{F|0}(t) = \frac{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t)}{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t) + (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)} \tag{3.10c}$$

$$\hat{\pi}_n^{H|0}(t) = \frac{(1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)}{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t) + (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)}. \tag{3.10d}$$

Using equations (3.10a) – (3.10d) in equation (3.8) the following result holds.

Lemma 1. *In case the sensor output is wrong ($r_n(t) = 1$), the maximum likelihood sensor state estimate is given by*

$$\hat{s}_n(t+1)|_{r_n(t)=1} = \begin{cases} H & \text{if } \hat{\pi}_n^H(t) > \frac{p_n^f(t)}{p_n^f(t)+p_n^h(t)} \\ F & \text{otherwise.} \end{cases} . \tag{3.11}$$

Similarly, in case the sensor output is correct ($r_n(t) = 0$), the maximum likelihood sensor state estimate is given by

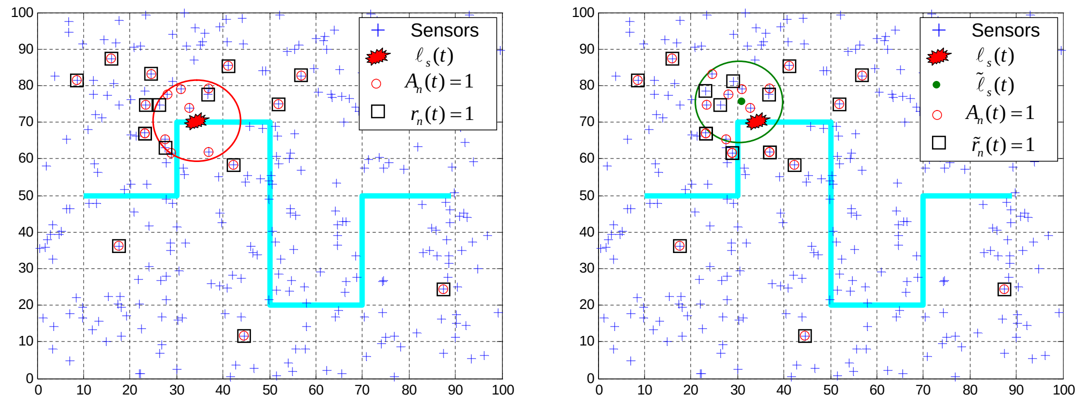
$$\hat{s}_n(t+1)|_{r_n(t)=0} = \begin{cases} F & \text{if } \hat{\pi}_n^H(t) < \frac{1-p_n^f(t)}{2-p_n^f(t)-p_n^h(t)} \\ H & \text{otherwise.} \end{cases} . \tag{3.12}$$

The derivation of equations (3.11) and (3.12) is relegated to Appendix A.

Corollary 1. *If $p_n^h(t) + p_n^f(t) = 1$, then the sensor health state estimators in equations (3.11) and (3.12) are simplified to*

$$\hat{s}_n(t+1)|_{r_n(t)=1} = \begin{cases} H & \text{if } \hat{\pi}_n^H(t) > p_n^f(t) \\ F & \text{otherwise.} \end{cases} \tag{3.13}$$

$$\hat{s}_n(t+1)|_{r_n(t)=0} = \begin{cases} F & \text{if } \hat{\pi}_n^H(t) < p_n^h(t) \\ H & \text{otherwise.} \end{cases} . \tag{3.14}$$



(a) Actual error signal $r_n(t)$.

(b) Estimated error signal $\tilde{r}_n(t)$.

Figure 3.4: Comparison between the actual and estimated sensor node error signal.

Given the error signal $r_n(t)$, it is evident that we only need to compute $\hat{\pi}_n^H(t)$, $p_n^h(t)$ and $p_n^f(t)$ in order to estimate the sensor health state with equation (3.11) or (3.12). Note, however, that $r_n(t)$ is not available because the actual target location is unknown. Thus, we use an estimate of the error signal, denoted as $\tilde{r}_n(t)$, by substituting $d_n(t)$ with $\tilde{d}_n(t)$, where $\tilde{d}_n(t) = \|\ell_n - \tilde{\ell}_s(t)\|$ denotes the distance between the n-th sensor and the estimated target location $\tilde{\ell}_s(t)$ provided by ftTRACK. The confidence we have on $\tilde{r}_n(t)$ depends on the uncertainty of the location $\tilde{\ell}_s(t)$. For instance, Figure 3.4 demonstrates how $\tilde{r}_n(t)$ is compared to the actual error signal $r_n(t)$. In this target tracking snapshot $\tilde{r}_n(t) \neq r_n(t)$ for 6 sensors. Thus, the estimated sensor state should be updated with caution because $\tilde{r}_n(t)$ may not be reliable for some sensors.

Next, we propose three different estimators for computing the sensor health state.

Simple Estimator

Assuming that the error signal $\tilde{r}_n(t)$ is always equal to 1 when the sensor is *Faulty* and always equal to 0 when the sensor is *Healthy*, then $p_n^f(t) = 1$ and $p_n^h(t) = 0$, $\forall t$. Therefore, if $\tilde{r}_n(t) = 1$ then equation (3.11) gives that the estimated state is H if $\hat{\pi}_n^H(t) > 1$ which is infeasible, thus in this case, the estimated state should always be F. Similarly, if $\tilde{r}_n(t) = 0$ then based on equation (3.12) the next state should be estimated as F if $\hat{\pi}_n^H(t) < 0$, which implies that the sensor state can never be estimated as *Faulty*. This leads to a simple estimation policy [83] given by

$$\hat{s}_n(t+1) = \begin{cases} H & \text{if } \tilde{r}_n(t) = 0 \\ F & \text{if } \tilde{r}_n(t) = 1 \end{cases}.$$

This estimator is quite intuitive and essentially, it says that if we fully trust the error signal, which is the basic assumption because $p_n^f(t) = 1$ and $p_n^h(t) = 0$, then the sensor health state is reliably estimated by $\tilde{r}_n(t)$. However, in the problem we are looking at, it is not possible to fully trust the error signal. For example, it is possible to have *correct* output when the sensor is faulty, e.g., if a sensor located far away from the target suffers a SA0 fault.

Static Estimator

The static estimator assumes that the MC has reached equilibrium and employs an estimate of the unknown steady state probability $\hat{\pi}_n^H$ in equation (3.11) or (3.12) to determine the sensor health state. The procedure is summarized in Algorithm 1 at Appendix B. The steady state probabilities are computed with

$$\begin{bmatrix} \hat{\pi}_n^F \\ \hat{\pi}_n^H \end{bmatrix} = \hat{C}_n^T(t) \begin{bmatrix} \hat{\pi}_n^F \\ \hat{\pi}_n^H \end{bmatrix}. \tag{3.15}$$

The transition probabilities $\hat{p}_n^{i,j}(t)$ in $\hat{C}_n(t)$ can be estimated simultaneously with target tracking by observing a sample of the estimated state $\hat{s}_n(t)$. The maximum likelihood estimate for the MC transition probabilities is given by

$$\hat{p}_n^{i,j}(t) = \frac{R_n^{i,j}(t)}{\sum_{k \in \{F,H\}} R_n^{i,k}(t)},\ i,j \in \{F,H\}, \tag{3.16}$$

where $R_n^{i,j}(t)$ is a counter that increments by one in case $\hat{s}_n(t-1) = i$ is followed by $\hat{s}_n(t) = j$. For example, if $\hat{s}_n(t) = \{H,F,F,F,H,H,H\}$ we easily see that $\hat{p}_n^{F,F}(t) = \hat{p}_n^{H,H}(t) = 2/3$, while $\hat{p}_n^{F,H}(t) = \hat{p}_n^{H,F}(t) = 1/3$.

Regarding the probabilities $p_n^h(t)$ and $p_n^f(t)$, in the general case $p_n^h(t) \neq 0$ because even in the fault-free case a healthy sensor that lies close to the *ROI* bound may have a *wrong* output due to noise in the measurements. On the other hand, $p_n^f(t)$ can be less than 1; for instance, if sensor n suffers from SA1 fault while the target passes through its neighbourhood, then $r_n(t) = 0$ even though the actual sensor state is *Faulty*. Similarly, if sensor n suffers from SA0 fault while the target moves far away, then again $r_n(t) = 0$. These cases indicate that $p_n^f(t) < 1$. Thus, for the static estimator, contrary to the simple estimator that uses $p_n^f(t) = 1$ and $p_n^h(t) = 0$, we calculate $p_n^h(t)$ and $p_n^f(t)$ for each sensor node according to the noise conditions in the field.

Lemma 2. *Assuming that $w_n(t) \sim \mathcal{N}(0, \sigma_w^2)$ is the noise disturbing the sensor measurements and the distance $d_n(t)$ is a normally distributed random variable $d_n(t) \sim \mathcal{N}(\tilde{d}_n(t), \sigma_d^2)$ the* *probabilities $p_n^h(t)$ and $p_n^f(t)$ are given by*

$$p_n^h(t) = \Big(1 - Q_w(t)\Big)\Big(1 - Q_d(t)\Big) + Q_w(t)Q_d(t) \tag{3.17}$$

$$p_n^f(t) = \Big(1 - Q_w(t)\Big)Q_d(t) + Q_w(t)\Big(1 - Q_d(t)\Big), \tag{3.18}$$

where $Q_w(t) = Q\Big(\frac{T-\mu_n(t)}{\sigma_w}\Big)$ with $\mu_n(t) = \frac{c}{1+\tilde{d}_n(t)^\zeta}$ and $Q_d(t) = Q\Big(\frac{R_I-\tilde{d}_n(t)}{\sigma_d}\Big)$.

The derivation of equations (3.17) and (3.18) is detailed in Appendix C.

The static estimator is expected to perform better compared to the simple estimator that fully trusts the error signal $\tilde{r}_n(t)$. However, the static estimator employs the estimated steady state probability $\hat{\pi}_n^H$, rather than the estimated state probability $\hat{\pi}_n^H(t)$ at time t, which leaves room for improvement using the following dynamic state estimation scheme.

Dynamic Estimator

The dynamic estimator, as opposed to the static estimator, considers the current measurement $\tilde{r}_n(t)$ not only for estimating the unknown sensor state, but also for updating the estimated sensor state probabilities according to

$$\begin{bmatrix} \hat{\pi}_n^F(t+1) \\ \hat{\pi}_n^H(t+1) \end{bmatrix} = \hat{C}_n^T(t) \begin{bmatrix} \hat{\pi}_n^{F|q}(t) \\ \hat{\pi}_n^{H|q}(t) \end{bmatrix}, \; q \in \{0,1\}, \tag{3.19}$$

where $\hat{C}_n^T(t)$ is given by equation (3.6) and the state transition probabilities are estimated using equation (3.16).

Essentially, all previous measurements $\tilde{r}_n(t)$ are encapsulated in the estimated sensor state probabilities and affect the future estimation steps. Thus, compared to the static estimator, the dynamic estimator is expected to improve the state estimation performance. However, our findings suggest that the performance of the dynamic estimator depends on the accuracy of the underlying target localization algorithm; see the discussion in Section 3.3.2. The estimation steps are listed in Algorithm 2 at Appendix B.

3.2.2 Target Localization

In principle, any target localization algorithm can be used in the ftTRACK architecture, including the simple CE algorithm [38] that estimates the target location as the arithmetic mean of the alarmed sensor locations

$$\hat{\ell}_s(t) = \Big(\frac{1}{P}\sum_{p=1}^{P} x_p, \frac{1}{P}\sum_{p=1}^{P} y_p\Big), \tag{3.20}$$

where (x_p, y_p), $p = 1, \ldots, P$ $(P \leq N)$ are the locations of the sensors that have been alarmed at time t, i.e., $A_p(t) = 1$.

However, such a localization algorithm is very sensitive to faults and may affect the performance of the ftTRACK architecture, especially when a large number of sensors fail. Thus, for the purposes of this thesis we adopted the SNAP algorithm [108] that is resilient to sensor faults. The SNAP algorithm fits well in our ftTRACK architecture because it is fault tolerant by itself and is therefore capable of handling some wrongly estimated sensor states during the ftTRACK computations. The SNAP algorithm consists of four components:

1. *Grid Formation:* The entire area is divided into a grid $\mathcal{G}$ with dimensions $R_x \times R_y$ and grid resolution g.

2. *Region of Coverage (ROC_n):* Given $\mathcal{G}$, the ROC_n of a sensor n is an area of grid cells around the sensor node.

3. *Matrix $\mathcal{L}(t)$ Construction:* If $A_n(t) = 1$, sensor n adds $+1$ inside the cells of matrix $\mathcal{L}(t)$ that correspond to ROC_n; otherwise ($A_n(t) = 0$), it adds -1. We calculate the "pseudo-likelihood" of a target found in each grid cell by summing the corresponding sensor contributions.

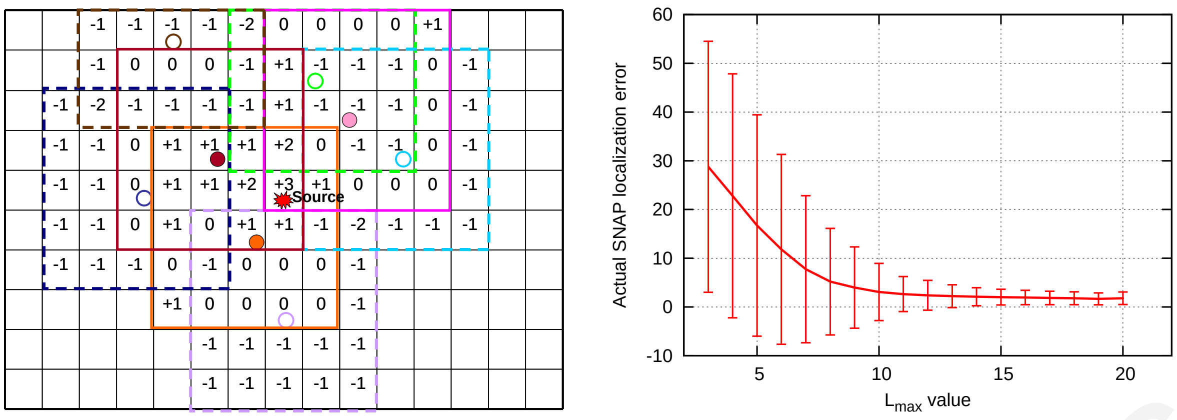
4. *Maximization:* The maximum value in the $\mathcal{L}(t)$ matrix, denoted $L_{max}(t)$, points to the estimated target location $\hat{\ell}_s(t)$.

An example using a square ROC_n is shown in Figure 3.5a. In this example, the target is correctly localized in the grid cell with $L_{max}(t) = +3$.

The target location estimates $\hat{\ell}_s(t)$ provided by the SNAP algorithm may occasionally deviate from the actual target path due to low sensor density in some parts of the field and/or high number of faulty sensors in the vicinity of the target. Our objective is to compensate for location estimates of diverse quality in order to penalize those estimates that are considered less accurate, while those that are more accurate are trusted more in the subsequent location smoothing process of ftTRACK. We estimate the localization error of SNAP $\hat{e}_s(t)$ using the following rule

$$\hat{e}_s(t) = \beta_L \frac{1}{L_{max}(t)}, \tag{3.21}$$

where $\beta_L > 0$ is a scaling factor. The rationale is that $L_{max}(t)$ reflects the uncertainty of the SNAP location estimate. For example, the higher the $L_{max}(t)$ value, the more accurate the SNAP location estimate is considered because it implies the contribution of a larger number of alarmed sensors.



(a) Example application of the SNAP algorithm. (b) L_{max} value as an indicator of the localization error.

Figure 3.5: SNAP localization and estimation of the associated location uncertainty.

The validity of equation (3.21) in the presence of faults is illustrated in Figure 3.5b, where the L_{max} values for 100,000 independent SNAP executions using a randomly generated target in each execution are plotted against the corresponding actual localization errors. We observe that this heuristic reflects the quality of the SNAP location estimates because the mean localization error and the associated standard deviation are both decreasing as the L_{max} value is increased. The effectiveness of equation (3.21) will also become evident later in the simulation results in Section 3.3, where we investigate the tracking error of the ftTRACK architecture.

3.2.3 Adaptive Particle Filter Smoothing

For the moving target, we use the target state and measurement model given by equations (3.5a) and (3.5b) and employ a particle filter to approximate the posterior Probability Density Function (PDF) of the target state given a series of measurements $p(X(t)|Y(1{:}t))$. The details of the centralized particle filter derivation can be found in [26]. Assuming that the PDF of the target state at time $t-1$ is available, the required posterior PDF is obtained in a recursive fashion through a prediction and an update step by using a set of N_p particles, i.e., random samples $\{X^i(t-1)\}_{i=1}^{N_p}$ with associated weights $\{\omega^i(t-1)\}_{i=1}^{N_p}$. The weights $\omega^i(t)$ are updated according to

$$\omega^i(t) = \omega^i(t-1)p(Y(t)|X^i(t)), \tag{3.22}$$

followed by a normalization step. We compute $p(Y(t)|X^i(t))$ using an approach similar to [42] in order to assign higher weight to those particles that are closer to the location estimate $\hat{\ell}_s(t)$ provided by the *Localization* component. In particular, we assume that $p(Y(t)|X^i(t))$ follows a Gaussian distribution

$$p(Y(t)|X^i(t)) = \frac{1}{\sqrt{2\pi}\sigma(t)} \exp\left[-\frac{(\bar{X}^i(t) - Y(t))^2}{2\sigma(t)^2}\right], \tag{3.23}$$

where $\bar{X}^i(t)$ denotes the position components in the state $X^i(t)$ of the i-th particle, which is updated using equation (3.5a). The parameter $\sigma(t)$ reflects the confidence in the measurement $Y(t)$, i.e., the uncertainty in the location estimate $\hat{\ell}_s(t)$. One option is to set this parameter to a constant value that represents the mean localization error of the SNAP algorithm [83, 84]. However, the SNAP estimates may sometimes be far from the actual target location, as discussed previously. For this reason, we use the estimated localization error of SNAP, computed at every time step using equation (3.21) in order to dynamically adapt the parameter $\sigma(t)$ by setting $\sigma(t) = \hat{e}_s(t)$.

Also, to avoid the particle degeneracy phenomenon, we employ a select with replacement linear time resampling algorithm for eliminating particles with low weights and moving them closer to particles with high weights, while keeping the computational overhead low [21].

The posterior PDF $p(X(t)|Y(1{:}t))$ is approximated as

$$p(X(t)|Y(1{:}t)) \approx \sum_{i=1}^{N_p} \omega^i(t)\delta(X(t) - X^i(t)), \tag{3.24}$$

where $\delta(\cdot)$ is the Dirac delta function. Finally, the filtered target location estimate is given by

$$\tilde{\ell}_s(t) \approx \sum_{i=1}^{N_p} \omega^i(t)\bar{X}^i(t). \tag{3.25}$$

The computation steps in the ftTRACK architecture are listed in Algorithm 3 at Appendix B.

3.3 Simulation Results

For the simulations we use a square 100×100 sensor field with $N = 600$ sensors, a single target that follows a staircase path sampled at $M = 180$ discrete time steps and we assume the measurement model of equation (3.2). A snapshot of the simulation setup is illustrated in Figure 3.6. The target follows a staircase trajectory (cyan) and at time t the actual target location $\ell_s(t)$ is shown in red, while the location estimated with ftTRACK $\tilde{\ell}_s(t)$ is shown in green. The alarmed sensors ($A_n(t) = 1$), either correctly due to the presence of the target or falsely due to noise or as a result of a fault, are marked with red circle. The sensors having an estimated *wrong* output ($\tilde{r}_n(t) = 1$) based on the circular *ROI* around $\tilde{\ell}_s(t)$ with radius $R_I = 10$m are marked with black square.

Faults are generated according to the four-state fault model presented in Section 3.1.3. By selecting the transition probabilities $p^{i,j},\ i, j \in \{RS, SA1, SA0, H\}$ we control which fault type is present in the field, as well as the percentage of faulty sensors, denoted α. For instance, if we set $p^{H,H} = 0.925$, $p^{RS,RS} = 0.7$, $p^{H,RS} = 0.075$ and $p^{RS,H} = 0.3$, while fixing the remaining transition probabilities to 0, then the steady state probabilities of the MC fault model are $[\pi_n^{RS}\ \pi_n^{SA1}\ \pi_n^{SA0}\ \pi_n^{H}]^T = [0.2\ 0\ 0\ 0.8]^T$. Thus, each sensor is expected to be at the RS state 20% of time or equivalently $\alpha = 20\%$ (i.e., around 120 out of 600 sensors will exhibit RS fault at each time step). Using $p^{H,H} = 1$ corresponds to the fault-free case ($\alpha = 0\%$). We point out that in our simulations we assume no prior knowledge about the transition probabilities of the four-state fault model, while the *Static* and the *Dynamic* estimators compute the transition probabilities in the underlying two-state MC simultaneously with tracking using equation (3.16).

We assess the performance of the *Simple*, *Static* and *Dynamic* sensor health state estimators, when employed in the ftTRACK architecture, and compare them in terms of the cumulative estimation error, defined as $\mathcal{E}_s = \frac{1}{NM}\sum_{t=1}^{M}\sum_{n=1}^{N}\epsilon_n(t)$, where

$$\epsilon_n(t) = \begin{cases} 0 & \text{if } \hat{s}_n(t) = H \text{ AND } s_n(t) = H \\ 0 & \text{if } \hat{s}_n(t) = F \text{ AND } s_n(t) \in \{RS, SA1, SA0\} \\ 1 & \text{if } \hat{s}_n(t) = F \text{ AND } s_n(t) = H \\ 1 & \text{if } \hat{s}_n(t) = H \text{ AND } s_n(t) \in \{RS, SA1, SA0\} \end{cases}.$$

In particular, we examine the estimation error $\mathcal{E}_s$ at the end of the target path (i.e., $t = M$). We also evaluate the ftTRACK architecture in terms of fault tolerance for different types of faults and report the tracking error pertaining to the whole target path defined as $\mathcal{E}_T = \frac{1}{M}\sum_{t=1}^{M}\|\ell_s(t) - \tilde{\ell}_s(t)\|$, i.e. the mean Euclidean distance between the actual and estimated target locations. Fault tolerance suggests that the tracking error degrades smoothly as the percentage of faulty sensors (α) is increased. The results on the sensor health state estimation error and the tracking error are averaged over 20 runs, where each run uses randomly deployed sensors.

3.3.1 ftTRACK with Centroid Estimator Localization

First, we investigate the performance of the ftTRACK architecture when the *Localization* component implements an algorithm that is sensitive to faults, such as CE that estimates the target location with equation (3.20). The *Smoothing* component is based on a particle filter with $N_p = 500$ particles, while we assume that the uncertainty of the CE location estimates

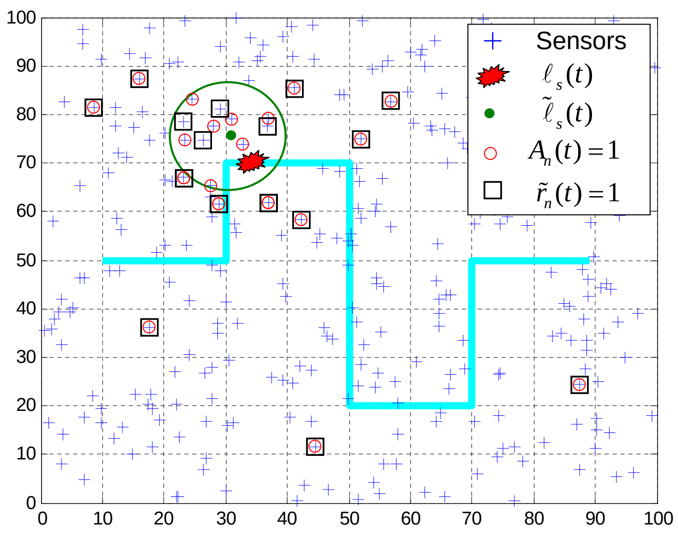


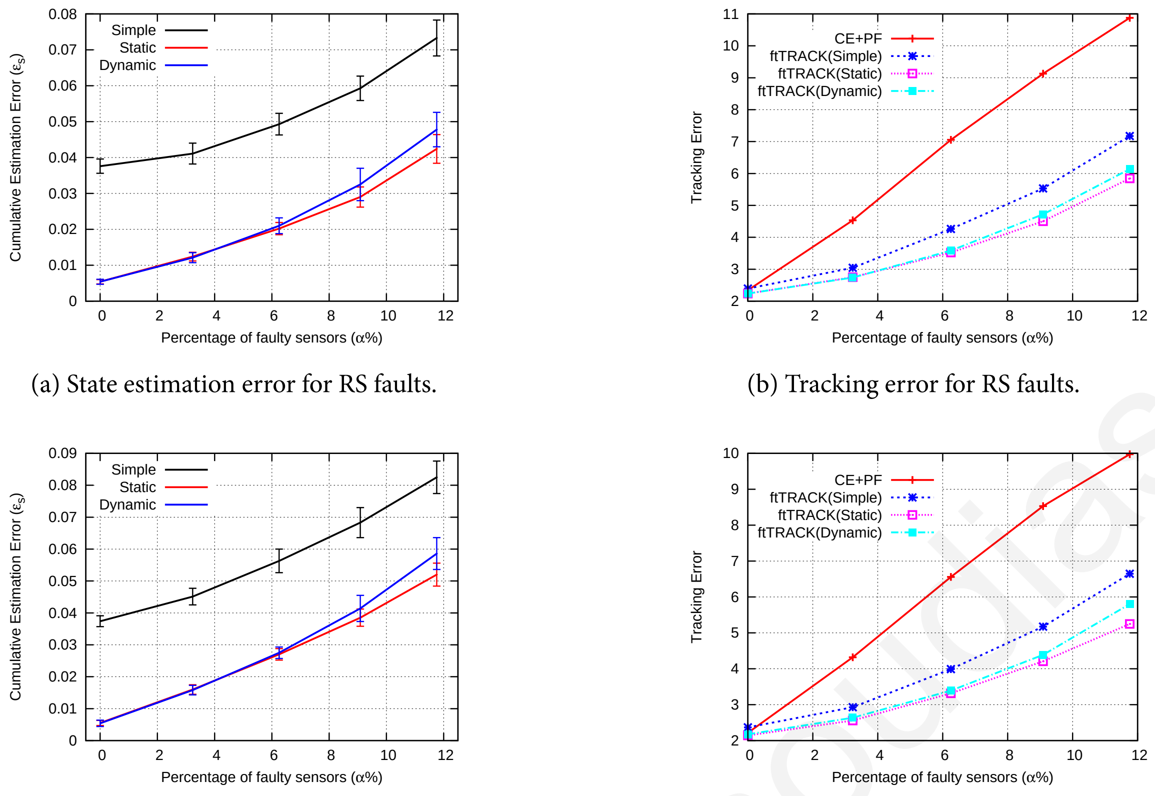
Figure 3.6: Simulation setup for the evaluation of ftTRACK.

is fixed to a constant value and selected $\sigma(t) = \hat{e}_s(t) = 4$, $\forall t$. Moreover, we assume $c = 3000$, $\zeta = 2$, $w_n \sim \mathcal{N}(0,1)$, $T = 5$ and $R_I = 24.5$ for the measurement model given by equation (3.2).

The simulation results with respect to the state estimation error and the tracking error, using different health state estimators, are shown in Figure 3.7 and Figure 3.8. We observe that the performance of the ftTRACK architecture under RS and SA1 faults is very similar and both the *Static* and the *Dynamic* estimators attain lower sensor state estimation error, compared to the *Simple* estimator, as shown in Figure 3.7a and Figure 3.7c. This indicates that considering the previously estimated sensor states in conjunction with the error signal $\tilde{r}_n(t)$ is beneficial for both types of faults and consequently this improves the tracking error; see Figure 3.7b and Figure 3.7d, respectively. For comparison, we also plot the tracking error of the original CE algorithm combined with the same particle filter that does not employ any sensor health estimator, denoted CE+PF. It is evident that the ftTRACK architecture improves the fault tolerance of the CE algorithm significantly and exhibits smoother accuracy degradation compared to CE+PF.

On the other hand, the CE algorithm is itself very resilient to SA0 faults because it takes into account only the alarmed sensors for estimating the target location. Thus, the tracking error of the CE+PF approach does not change much as α is increased. For this type of fault using the *Static* or *Dynamic* estimator in the ftTRACK architecture leads to lower sensor state estimation error (Figure 3.8a), while the improvement in the tracking accuracy is marginal, as shown in Figure 3.8b.

The observations above are also confirmed in a scenario with mixed faults, as shown in



(a) State estimation error for RS faults. (b) Tracking error for RS faults.

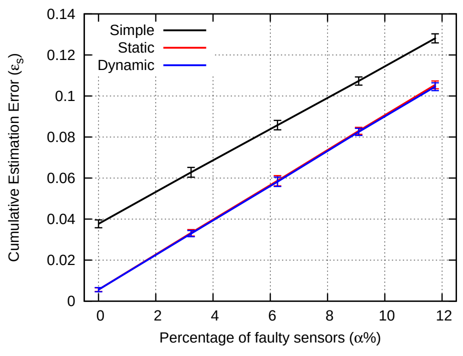
(c) State estimation error for SA1 faults. (d) Tracking error for SA1 faults.

Figure 3.7: Performance for RS and SA1 faults using the CE algorithm in ftTRACK.

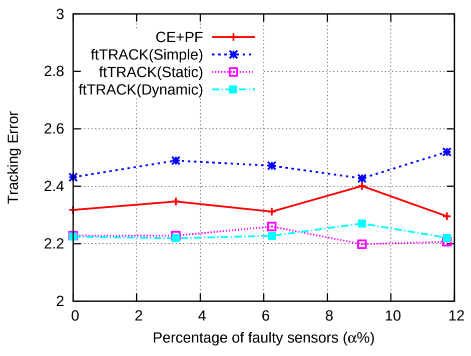
Figure 3.8c and Figure 3.8d. We assume that there are no transitions between faulty states and we fix the transition probabilities as $p^{RS,H} = 0.1$, $p^{SA1,H} = 0.4$ and $p^{SA0,H} = 0.4$, while we appropriately vary the probabilities $p^{H,H} \in [0.96,\ 1]$, $p^{H,RS} \in [0,\ 0.008]$ and $p^{H,j} \in [0,\ 0.016]$, $j \in \{SA1, SA0\}$. In this way, we simulate the case where different faults are present in the field at the same time, while a sensor that suffers from a RS fault is much less likely to recover, compared to SA1 and SA0 faults.

3.3.2 ftTRACK with SNAP Localization

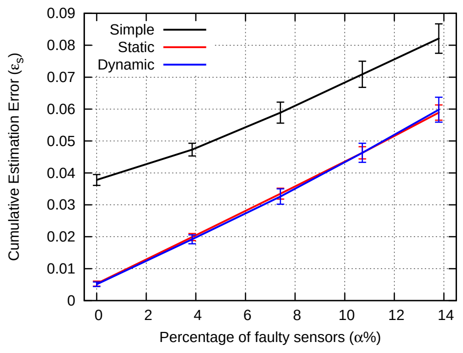
When the CE algorithm is used, the ftTRACK architecture provides adequate tracking performance for a small number of faulty sensors. To get the full benefit of ftTRACK we propose to employ a localization algorithm that is fault tolerant itself, such as SNAP. In this way, we can further improve the tracking performance, especially in case a large number of faulty sensors are present in the field.



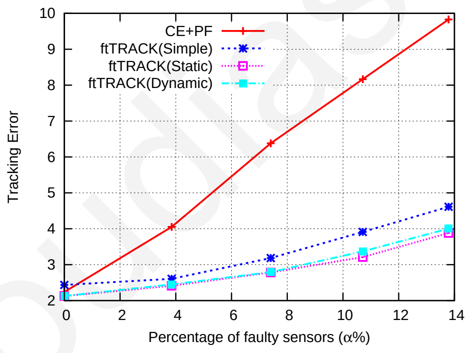
(a) State estimation error for SA0 faults.



(b) Tracking error for SA0 faults.



(c) State estimation error for mixed faults.



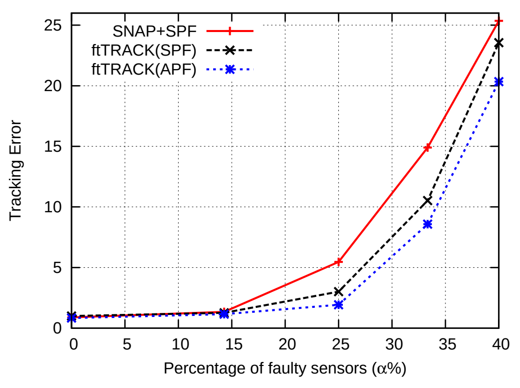
(d) Tracking error for mixed faults.

Figure 3.8: Performance for SA0 and mixed faults using the CE algorithm in ftTRACK.

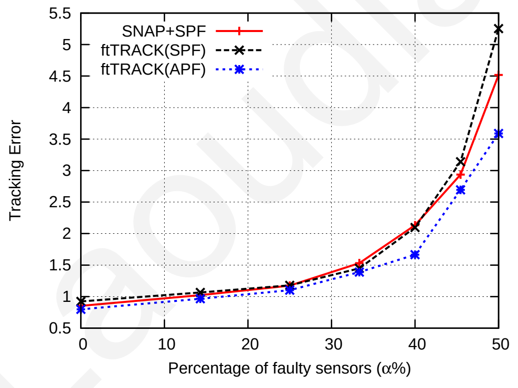
Evaluation of the Adaptive Particle Filter

First, we demonstrate the effectiveness of the Particle Filter (PF) that is employed in the ftTRACK architecture for smoothing the location estimates derived with the SNAP localization algorithm. In these simulations we assume the measurement model of equation (3.2) with $c = 5000$, $\zeta = 2$, $w_n \sim \mathcal{N}(0,1)$, $T = 50$ and $R_I = 10$. We consider two variants of the ftTRACK architecture that differ only with respect to the location smoothing component. The first variant, denoted ftTRACK(SPF), uses a Standard PF where the uncertainty of the SNAP estimates is fixed to a constant value and we selected $\sigma(t) = 2,\ \forall t$ [83,84]. The second variant, referred to as Adaptive PF and denoted ftTRACK(APF), adapts the uncertainty in the current SNAP estimate dynamically using equation (3.21) with $\beta_L = 20$. Both ftTRACK variants employ the *Simple* estimator for estimating the sensors' health states. For comparison, we also plot the performance of the original SNAP algorithm combined with the Standard PF that does not employ any sensor health estimation scheme, denoted SNAP+SPF. Note that in all three PF implementations we have used $N_p = 500$ particles for approximating the posterior PDF of the target state.

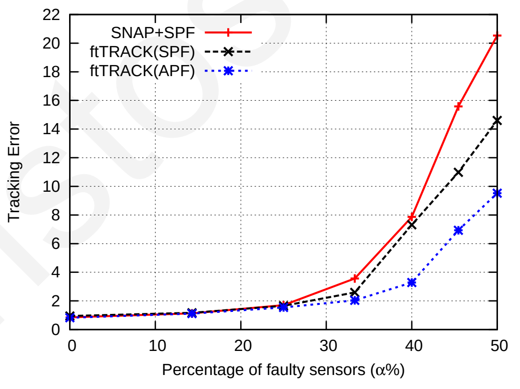
The tracking error for different types of faults is plotted in Figure 3.9. Our first observation is that the proposed ftTRACK architecture improves the fault tolerance of the original SNAP algorithm, mainly due to sensor health state estimation. Moreover, we observe that the variant based on the Adaptive PF provides considerable improvement over the Standard PF approach, especially when RS or SA0 faults are considered. For SA1 faults the tracking error is only slightly improved because the SNAP algorithm is extremely robust to this type of fault. In general, the adaptive rule in equation (3.21) seems to handle well the different types of faults and can adequately quantify the uncertainty in the SNAP location estimates for varying number of faulty sensors in the field. Thus, we employ the Adaptive PF in the ftTRACK computations hereafter.



(a) Reverse Status faults.



(b) Stuck-At-1 faults.



(c) Stuck-At-0 faults.

Figure 3.9: Effectiveness of the Adaptive Particle Filter in the ftTRACK architecture.

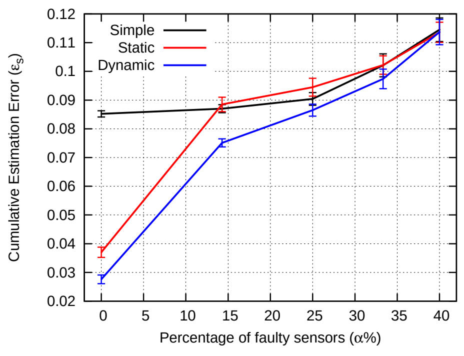
Performance Evaluation of ftTRACK

Next, we investigate the state estimation error and the tracking error when each of the sensor health state estimators are used in ftTRACK. We are interested in the performance of ftTRACK in the presence of faults and higher noise, thus for the following simulations we assume the same measurement model as before with $w_n \sim \mathcal{N}(0, 1000)$. For comparison we plot the tracking error of SNAP+SPF, which is not using any sensor state estimator, as shown in Figure 3.10 and Figure 3.11. We also consider a representative Trust Index algorithm (TI) [141] combined with Standard PF, denoted as TI+SPF. Note that the TI algorithm tries to reduce the impact of faulty sensors by reducing their trust indices during target tracking according to an exponential function.

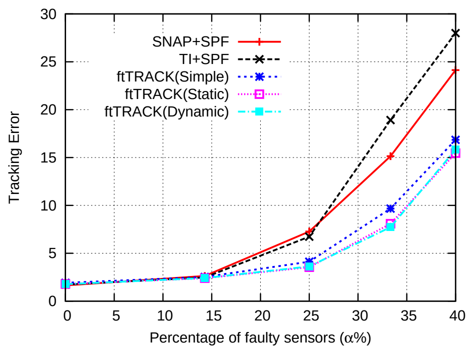
For RS faults, we observe that the sensor state estimation error $\mathcal{E}_s$ when $\alpha < 25\%$ is much higher for the *Simple* estimator, compared to the *Dynamic* estimator; see Figure 3.10a. Especially in the fault-free scenario ($\alpha = 0\%$), $\mathcal{E}_s$ is three times larger because the *Simple* estimator incorrectly estimates the state of several healthy sensors as faulty, while those sensors may sometimes have *wrong* output due to measurement noise. However, the superiority of the *Dynamic* estimator is not reflected in the tracking error because the SNAP algorithm is fault tolerant and effectively masks these incorrect sensor state estimates. Using the *Static* or the *Dynamic* estimator provides additional improvement only when the percentage of faulty sensors increases beyond 25%, as shown in Figure 3.10b. In general, the ftTRACK architecture proves to be very robust to RS faults and outperforms both the SNAP+SPF and the TI+SPF approaches in terms of the tracking error for $\alpha > 15\%$, which demonstrates the effectiveness of sensor health state estimation in the tracking process.

In the case of SA1 faults, the *Dynamic* estimator always achieves lower $\mathcal{E}_s$ compared to the *Static* estimator and is better than the *Simple* estimator when $\alpha < 30\%$ (Figure 3.10c). However, the SNAP algorithm compensates the incorrect sensor state estimates and the ftTRACK architecture provides similar tracking accuracy for all three sensor state estimators; see Figure 3.10d. The *Simple* estimator has high $\mathcal{E}_s$ when few sensors are faulty due to the measurement noise, but as the number of faulty sensors grows larger trusting only the error signal $\tilde{r}_n(t)$ is a good strategy and this is reflected in the tracking error of the ftTRACK architecture when more than 40% of the sensors are faulty. Note that the SNAP algorithm is by default very resilient to this type of fault because several SA1 sensors must be concentrated far from the actual target location in order to introduce error in the SNAP localization.

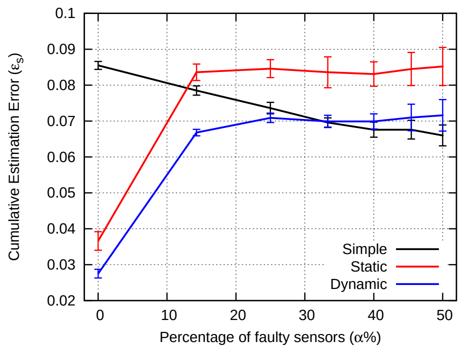
When SA0 faults are present in the field, $\mathcal{E}_s$ is significantly increased for all three estimators compared to RS and SA1 faults; see Figure 3.11a. This is expected because there are several sensors located far from the target path that exhibit SA0 fault and these sensors are estimated as *Healthy* because according to their locations with respect to the target location they should not be alarmed. The state of these sensors can only be estimated correctly if the target passes by their neighbourhood, so that they provide *wrong* output for some time. These sensors do not affect the performance of the ftTRACK architecture because they would anyway remain



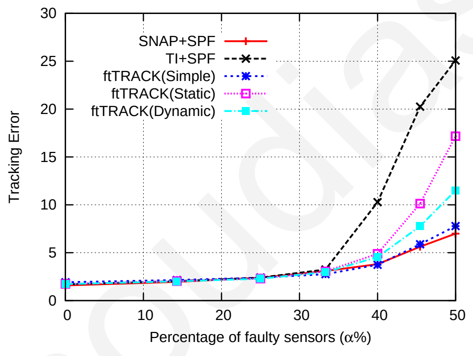
(a) State estimation error for RS faults.



(b) Tracking error for RS faults.



(c) State estimation error for SA1 faults.



(d) Tracking error for SA1 faults.

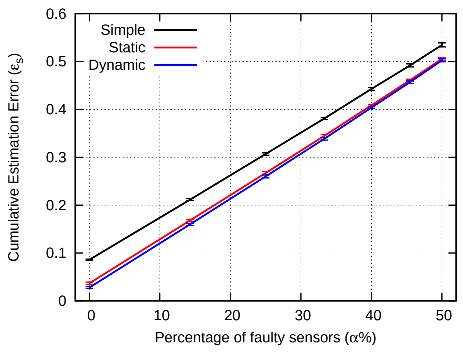
Figure 3.10: Performance under RS and SA1 faults using the SNAP algorithm in ftTRACK.

silent under normal operation. The tracking accuracy can only be affected by SA0 sensors falling inside the target *ROI* because such sensors falsely indicate that the target is not located in their vicinity. We point out that these faulty sensors are correctly identified by our sensor state estimators. In particular, the *Static* and the *Dynamic* estimators can better handle this type of fault when the measurement noise is high and the tracking error of the ftTRACK architecture is further decreased when $\alpha > 25\%$, compared to the *Simple* estimator.

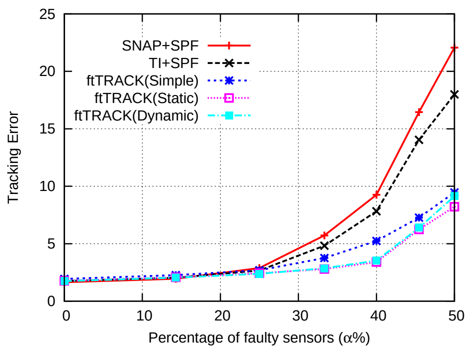
Finally, to generate mixed faults we assume that there are no transitions between faulty states and we fix the transition probabilities as $p^{RS,H} = 0.1$, $p^{SA1,H} = 0.4$ and $p^{SA0,H} = 0.4$, while we appropriately vary the probabilities $p^{H,H} \in [0.75,\ 1]$, $p^{H,RS} \in [0,\ 0.05]$ and $p^{H,j} \in [0,\ 0.1]$, $j \in \{SA1, SA0\}$. In this case, the ftTRACK architecture reduces the tracking error significantly, as shown in Figure 3.11c and Figure 3.11d.

Effect of Permanent Faults

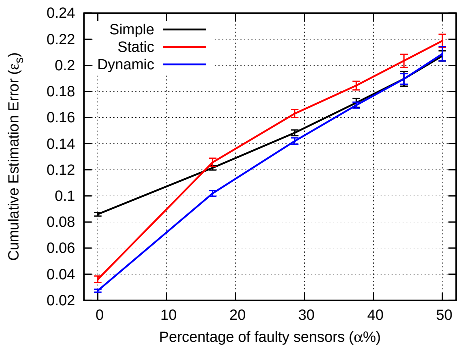
In the simulation results presented so far, we have considered faults that come and go in a temporary fashion. However, some failures, including software bugs, too high/low detection threshold or a dirty sensor, may also manifest themselves as permanent faults. Different types



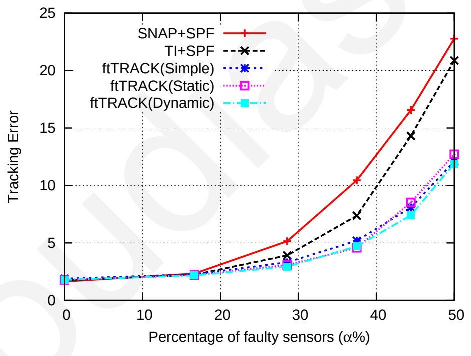
(a) State estimation error for SA0 faults.



(b) Tracking error for SA0 faults.



(c) State estimation error for mixed faults.



(d) Tracking error for mixed faults.

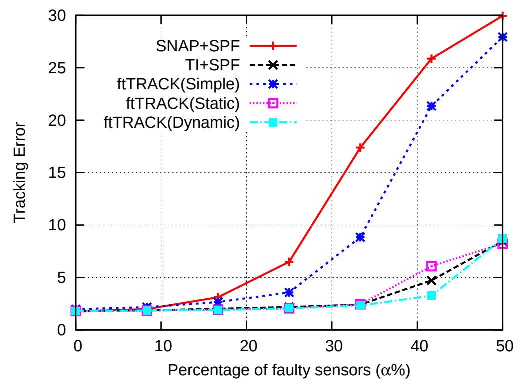
Figure 3.11: Performance under SA0 and mixed faults using the SNAP algorithm in ftTRACK.

of permanent faults can be generated with our fault model by fixing the transition probability of the MC as $p^{i,i} = 1,\ i \in \{RS, SA1, SA0\}$. The tracking error of ftTRACK with the SNAP localization algorithm, when permanent RS faults are present, is depicted in Figure 3.12a. An important observation is that the *Simple* estimator is severely affected by permanent faults and the tracking error increases sharply, compared to the *Static* and *Dynamic* estimators. This behaviour is attributed to the fact that the *Simple* estimator does not consider the previously estimated sensor state. Thus, the errors in the estimated source locations make the estimator oscillate between the *healthy* and *faulty* state, even though the faults are actually permanent.

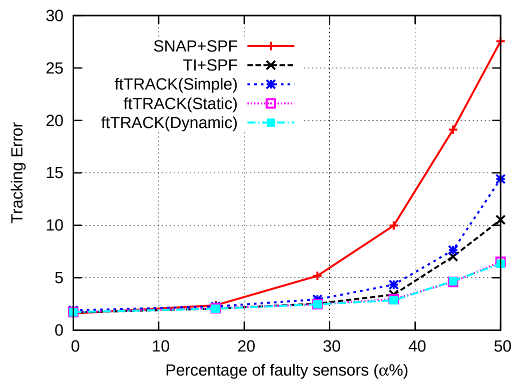
On the other hand, permanent faults can be handled effectively using the *Static* and *Dynamic* estimators in ftTRACK. This is also confirmed in a more realistic scenario where a subset of randomly selected sensors exhibit temporary mixed faults, while another smaller subset suffers from permanent RS faults, as shown in Figure 3.12b.

Effect of Spatially Correlated Faults

The results above indicate that the ftTRACK architecture is capable of tracking a target reliably even when a large number of sensors fail, either temporarily and/or permanently.



(a) Permanent RS faults.



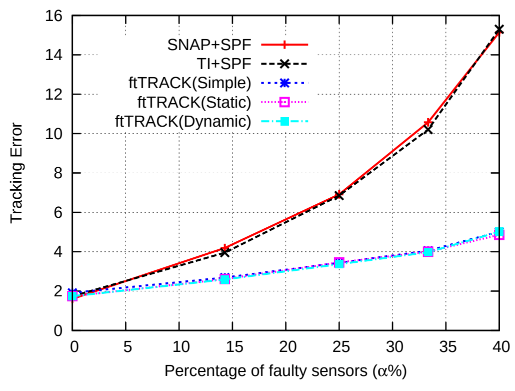
(b) Temporary mixed and permanent RS faults.

Figure 3.12: Performance of ftTRACK in the presence of permanent faults.

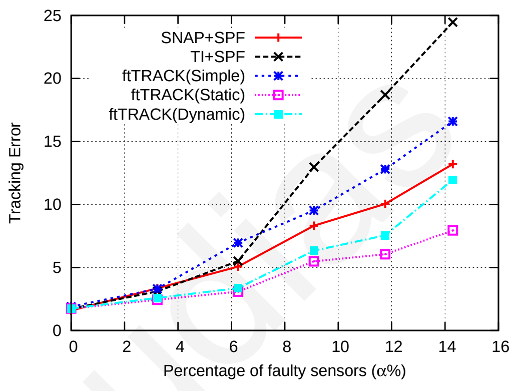
So far, we have assumed that the sensors' health states are spatially independent. This is sufficient for modelling different types of failures that appear to be uncorrelated in real WSN applications, however, in some cases the sensor faults may exhibit strong spatial correlation. For instance, if there is a fire in a part of the field, then several sensors in that area may suffer from stuck-at fault due to board overheating. In another scenario, an intruder may deploy a number of small decoy sources far from its actual location [106]. Thus, all sensors around the decoy sources will become alarmed, i.e., exhibit SA1 fault. This is similar to the case reported from the 1980s, where the rebels in Afghanistan threw rabbits over base fences that caused the motion detectors to generate a series of false alarms [132].

To simulate spatially correlated faults we assume that the sensor field is divided into 10×10 non-overlapping subregions, while each subregion contains 6 sensors on average. All sensors in a subregion form a cluster and we have randomly selected one sensor as the head. We used our fault model to control the state of the cluster heads, while all other sensors in each cluster have the same health state with the corresponding cluster head at each time step. The tracking error for increasing percentage of faulty sensors in the field is depicted in Figure 3.13. We observe that ftTRACK is very resilient to spatially correlated SA0 faults and all estimators perform similarly, while the ftTRACK architecture outperforms both the SNAP+SPF and TI+SPF approaches (Figure 3.13a). Note that these SA0 faults affect the tracking task only when the target moves through a faulty sensor cluster. On the other hand, spatially correlated SA1 faults have a greater effect on the tracking performance because they introduce high errors in the localization algorithm, i.e., the estimated target location may diverge towards the falsely alarmed sensors in a faulty cluster. In this case, TI+SPF performs poorly, while ftTRACK is more fault tolerant especially if the *Static* or *Dynamic* estimators are employed, instead of the *Simple* estimator, as shown in Figure 3.13b.

We point out that the estimators in this work do not consider spatially correlated faults explicitly and the health state is estimated independently for each sensor according to the associated error signal. As part of our future work, we plan to investigate sensor state estimators that will take this correlation into account.



(a) Spatially correlated SA0 faults.



(b) Spatially correlated SA1 faults.

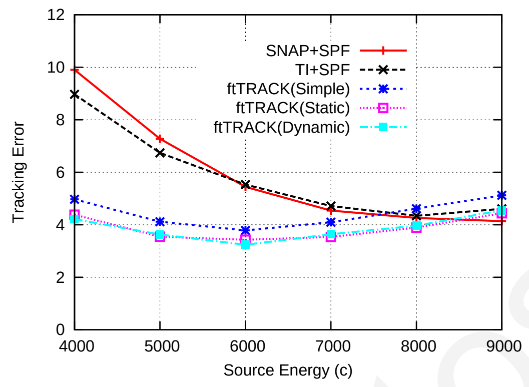
Figure 3.13: Performance of ftTRACK in the presence of spatially correlated faults.

Effect of Variable Source Energy

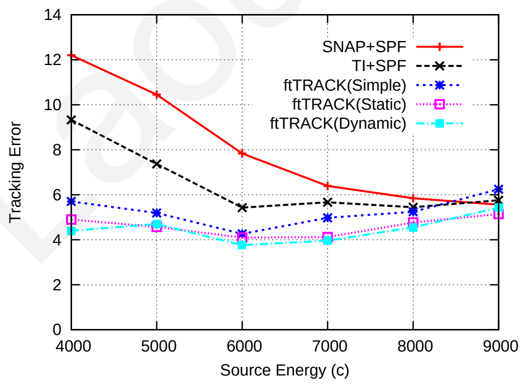
We now investigate another interesting scenario in which the source that moves through the field has an unknown energy level c, which is different than the expected one. Recall that c determines the *ROI* of the source, because the radius of the *ROI* is $R_I = \sqrt[\zeta]{c/T - 1}$, while the *ROI* size affects the likelihood matrix of SNAP, as well as the error signal used by the sensor health estimator. To investigate how the value of c affects the tracking performance of ftTRACK, we perform additional simulations by keeping constant the anticipated energy value within the ftTRACK computations (i.e., $c' = 5000$), while varying the real source energy $c \in [4000, 9000]$.

The real source energy c affects the number of sensors around the source that are actually alarmed. When $c > c'$, additional sensors that are further away from the source can also get correctly alarmed, however, the sensor health state estimator eventually estimates these alarmed sensors as faulty because they fall outside the anticipated *ROI*. These sensors are simply excluded from the tracking task, while the remaining sensors inside the anticipated *ROI* continue to be considered as healthy. Thus, the accuracy of the SNAP localization algorithm is not affected and the smooth operation of ftTRACK is preserved. On the other hand, when $c < c'$, fewer sensors become correctly alarmed and the performance of SNAP may be degraded. Moreover, there are several sensors falling inside the anticipated *ROI*, which are estimated as faulty, even though they correctly remain non alarmed. Thus, these wrong sensor state estimates may affect the subsequent tracking steps by excluding sensors that are healthy.

The tracking error in the presence of temporary faults is plotted in Figure 3.14a and Figure 3.14b assuming 25% RS faults and 38% mixed faults, respectively. We observe that ftTRACK is resilient to energy variations, even in the presence of faults. The plots indicate that it is better to be conservative in the value of the anticipated source energy in order to be able to handle sources with different energy effectively. Moreover, this highly desirable behaviour comes without extra cost, as no modifications are required in the ftTRACK architecture. As a point for future work, we note that a better approach would be to estimate the source energy online, i.e., simultaneously with tracking. In this way, the sensor state estimator will use the appropriate value for the *ROI* radius within ftTRACK to match the actual source energy.



(a) Temporary RS faults ($\alpha = 25\%$).



(b) Temporary mixed faults ($\alpha = 38\%$).

Figure 3.14: Performance of ftTRACK for variable source energy.

Discussion on Estimators

Looking at the simulation results, in conjunction with the results in Section 3.3.1 obtained with the CE localization algorithm, we observe that in most cases the *Dynamic* estimator does not perform better than the *Static* estimator. In order to shed light on this behaviour, we carefully examined several tracking scenarios in our simulation setup. Our findings suggest that the *Dynamic* estimator is more sensitive to the wrongly estimated source location. In other words, we noticed that when the error in the estimated source location is high, this has a greater effect on the *Dynamic* estimator. This is because the *Dynamic* estimator uses the current error signal $\tilde{r}_n(t)$, which is determined according to the estimated source location, to predict the state probabilities of the MC for the next time step; see equation (3.19), as well as steps 11 and 19 of Algorithm 2 in Appendix B. On the other hand, the *Static* estimator relies on the steady state probabilities that are not immediately affected by the error signal.

This is confirmed by our simulation results. In particular, when the CE localization algorithm is employed, then the estimated source location is not very accurate, e.g., for RS and SA1 faults (see red line in Figure 3.7b, Figure 3.7d) and the *Dynamic* estimator performs slightly worse than the *Static* estimator in terms of the state estimation error, thus leading to higher tracking error of ftTRACK when more sensors fail (Figure 3.7a–Figure 3.7d). On the other hand, the SNAP localization algorithm is more accurate, e.g., under SA1 faults (see red line in Figure 3.10d) and in this case the *Dynamic* estimator is better, leading to lower tracking error when a large number of sensors have failed.

3.4 Distributed ftTRACK

3.4.1 Preliminaries

Following our developments in the context of single source tracking in a centralized network architecture, our objective now is to enhance ftTRACK for enabling multiple source tracking in a distributed manner. Three issues need to be addressed for supporting such a distributed ftTRACK variant, namely source identification, distributed localization and distributed sensor health state estimation. Next, we present efficient solutions for distributed source identification and localization, while we leave the challenging problem of distributed sensor health state estimation as part of our future work.

For the purposes of distributed multiple source tracking, we assume that each sensor node has a limited communication range R_S and a set of K point event sources are moving inside the sensor field $\mathcal{A}$ according to a piecewise linear pattern. At time t, the k-th source is located at position $\ell_k(t) = (x_k(t), y_k(t))$ inside $\mathcal{A}$. In the distributed environment, $z_n(t)$ is the received signal of sensor n located at (x_n, y_n) and is given by the sum of the signals from all sources at the sensor location

$$z_n(t) = \sum_{k=1}^{K} s_{n,k}(t) + w_n(t), \tag{3.26}$$

where $w_n(t)$ is additive white Gaussian noise, $w_n(t) \sim \mathcal{N}(0, \sigma_w^2)$ for $n = 1, \dots, N$ and $t = 1, \dots, M$. In accordance with the single source signal model given by equation (3.2), the signal from each source k is given by

$$s_{n,k}(t) = \frac{c_k}{1 + d_{n,k}(t)^{\zeta}}, \tag{3.27}$$

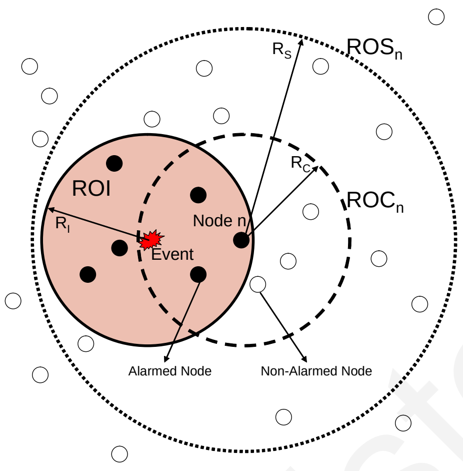
where c_k is some constant characterizing the k-th source and $d_{n,k}(t)$ is the radial distance of sensor node n from source k at time t

$$d_{n,k}(t) = \sqrt{(x_n - x_k(t))^2 + (y_n - y_k(t))^2}. \tag{3.28}$$

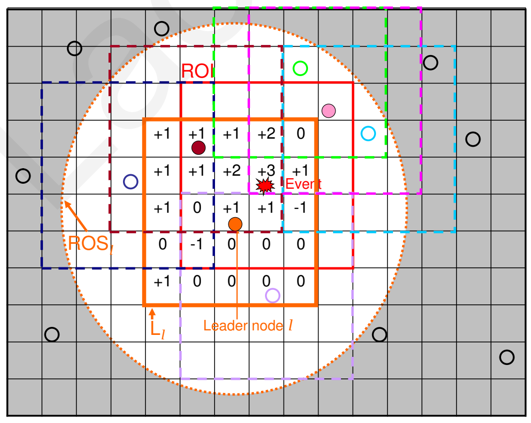
Moreover, following the definitions of the source *ROI* and sensor node ROC_n in Section 3.1.1 we now define the *Region of Subscription (ROS)* of node n that is needed in the context of distributed computation.

Definition 5. *Region of Subscription (ROS_n) of sensor node n, is the area around the sensor node n location inside which the sensor node needs to subscribe for information from all other sensor nodes which are relevant to the application considered.*

We also refer to this region as the *neighborhood* of sensor node n; see Figure 3.15a.



(a) Region of Subscription (ROS_n).



(b) Example application of dSNAP.

Figure 3.15: Computations in the distributed ftTRACK architecture.

3.4.2 Source Identification

We now describe an efficient source identification scheme for detecting and correctly counting the number of sources in the distributed architecture [107]. In a nutshell, during the identification phase a subset of the sensor nodes are elected as leaders and our objective is that the number of elected leaders will correspond (or at least be close) to the actual number of sources in the field. Note that the number of sources is not known in advance, so we cannot make any assumption regarding the number of sources or the time that they enter the field.

Sensor node n can become a leader in two possible ways:

1. Through a Distributed Fault Tolerant Leader Election Protocol (D-FTLEP), where node n did not hear any neighbour becoming a leader and the majority of its neighbours are alarmed.

2. Through leadership transfer, where node n receives a message from a neighbouring node, which is currently a leader, indicating that a source is moving closer to n and thus it should become the new leader.

D-FTLEP

Initially, each alarmed sensor node n broadcasts an *ALARM* message inside its ROS_n and subsequently it computes the following function F_n using the received information,

$$F_n = \sum_{m \in ROS_n} b_m, \tag{3.29}$$

where

$$b_m = \begin{cases} +1 & \text{if } A_m(t) = 1 \\ -1 & \text{if } A_m(t) = 0 \end{cases}. \tag{3.30}$$

Note that for the sensor nodes inside ROS_n for which no message is received, a non-alarm status ($A_m(t) = 0$) is implied. If $F_n > 0$ for *at least one* alarmed sensor node, then at least one source has been detected.

The use of the value F_n is an essential part of the leader election protocol for the two following reasons. First, it causes the leader node to be elected close to the actual source location because sensor nodes closer to the source have (on average) more alarmed neighbours. Note that this is achieved using only binary information from the sensors. Second, by using a bounded ± 1 contribution from the sensors our protocol can handle a percentage of erroneous observations provided by faulty nodes.

Next, sensor node n with $F_n > 0$ waits for a period that is given by a strictly increasing function $h(x)$. If during this period, n does not receive a *LEADER* message with value $f \geq F_n$ it implies that n is the node with the most alarmed neighbours and is likely located close to the source. Thus, it becomes the node leader l and broadcasts a *LEADER* message to its neighbours.

If node n receives a *LEADER* message with value $f \geq F_n$, then it is "suppressed" meaning that for a certain period it cannot become a leader itself. Note that according to the algorithm a sensor m outside the ROS_l of an elected leader with $F_m > 0$ may also become leader if it has the maximum value among its neighbours (excluding the ones in the ROS_l of the already elected leader which are suppressed). This turns out to be an important feature of the algorithm for correctly counting sources that are located close to each other (e.g., targets with crossing trajectories). Also, note that for every source the number of messages required for the leader election protocol is linear with respect to the cardinality of the leader's neighbourhood.

A leader may initiate and maintain a track as described above. A leader should also terminate a track when a source moves away and it is not clear who the next leader is going to be. Therefore, a leader l who did not determine the next leader and for three consecutive steps computes a value $F_n \leq 0$ simply terminates the track. The steps of D-FTLEP are listed in Algorithm 4 at Appendix D.

Leadership Transfer

Alternatively, a current leader can determine whether a source has moved closer to one of its neighbours through the particle filter algorithm running in the *Smoothing* component of ftTRACK. Thus, it can inform this neighbour that it should become the next leader, i.e., transfer the leadership.

At every iteration, the particle filter produces a prediction of the next state of the target, including the target's position and velocity; see step 3 of Algorithm 3 in Appendix B. If the predicted position is closer to a neighbor n of the current leader l, then the leader sends a message to n with the current state of the filter, so that n will be the new leader to continue the track. At this point, the new leader will also broadcast a *LEADER* message in order to suppress all of its neighbours from becoming leaders.

Corollary 2. *In order to guarantee that a set of consecutive leaders can continuously track a moving target, the separation distance between any two consecutive leader nodes* $r_{sep}(l_t, l_{t+1})$ *should be less than* $2R_C$.

Proof. If the two leaders (l_1 and l_2) are located at a distance greater than $2R_C$, their *ROCs* do not overlap. When the source is about to exit from the *ROC* of l_1, l_1 informs l_2 that it will become the new leader. However, at this point, it is also possible that the source may change direction and as a result is will never enter the *ROC* of l_2. In this case, l_2 will lose track of the source, as shown in Figure 3.16. □

In situations where the distance between two consecutive leaders is more than $2R_C$ or when the next source position is "wrongly" computed, then the next leader might be elected outside of the *ROI* of the source and as a result the track is falsely abandoned (e.g., when the majority of the nodes inside the leader neighbourhood become non-alarmed). At this point a new track will be initiated by a newly elected leader following the D-FTLEP algorithm described previously.

Another situation where a track is falsely terminated is when the paths of two sources cross each other and their separation distance is $r_{sep} \leq 2R_C$. In this case, the localization algorithm "sees" only a single source and consequently only a single target can be tracked, thus the other is abandoned. When the two sources move sufficiently apart, the D-FTLEP can once again detect the source that has been abandoned and a new track is re-initiated.

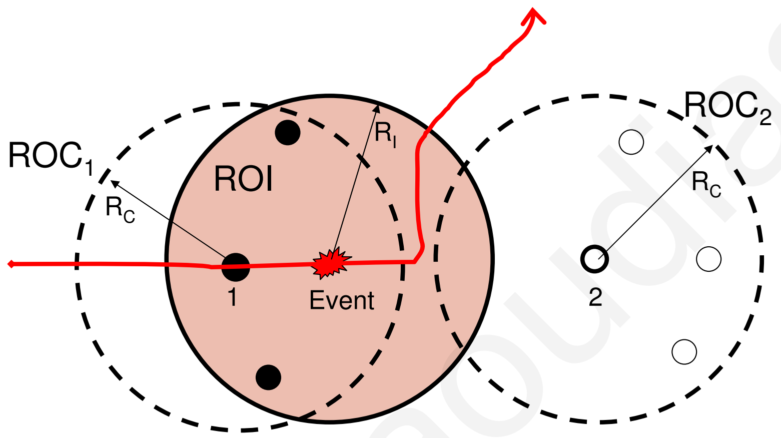


Figure 3.16: Target tracking scenario where next elected leader loses target.

3.4.3 Distributed Source Localization

The SNAP algorithm described in Section 3.2.2 is a centralized localization algorithm [108]. Motivated by that algorithm, we now present a variant, denoted dSNAP, which is adapted for distributed network architectures [104, 107].

The dSNAP algorithm assumes a quantized field, thus the entire area is divided in a grid $\mathcal{G}$ with $G \times G$ cells and grid resolution g. Depending on the resolution a cell may contain multiple sensors or no sensors at all. The position index of each node $n = 1, \cdots, N$ is denoted by (X_n, Y_n), where $X_n, Y_n \in \{1, 2, \ldots, G\}$ are the *discrete* position indices related to the real-valued position of the node (x_n, y_n) by $X_n = \left\lceil \frac{x_n}{g} \right\rceil, Y_n = \left\lceil \frac{y_n}{g} \right\rceil$.

The algorithm is run by the leader node l which needs to collect the alarm status of all sensors inside its ROS_l. Note that this information is already available from the identification phase. Using this information, the leader constructs the scoring matrix $\mathcal{L}_l$ over a sub-grid $\mathcal{G}_l$ around its location. The maximum value of $\mathcal{L}_l$ points to the estimated position of the source. Next, we present the details of the algorithm.

The leader node $l, l \in \{1, \cdots, N\}$ is associated with $\mathcal{G}_l$, a sub-grid of $\mathcal{G}$ with $G_l \times G_l$ cells, centred around its location (X_l, Y_l). The size of the sub-grid $G_l = \left\lfloor \frac{R_S}{g} \right\rfloor + 1$ depends on the size of the ROS_l and the grid resolution g. Furthermore, node l defines a $G_l \times G_l$ scoring matrix $\mathcal{L}_l$ where each element (i, j) of $\mathcal{L}_l$ corresponds to a cell (u, v) of $\mathcal{G}_l$. This relation is given by a mapping $\mathcal{M}_l : \mathcal{G}_l \to L_l$, thus

$$\mathcal{M}_l\left([u,v]^T\right) = \left[u - X_l + \left\lceil \frac{G_l}{2} \right\rceil, v - Y_l + \left\lceil \frac{G_l}{2} \right\rceil\right]^T,$$

where $u, v \in \{1, \cdots, G\}$ and $i, j \in \{1, \cdots, G_l\}$. For every element of $\mathcal{L}_l$, the leader adds the contribution of each sensor that has the corresponding cell in its ROC. The contributions can be ± 1 depending on the sensor's status: positive observation $+1$ when $A_m(t) = 1$ and negative observation -1 when $A_m(t) = 0$. Specifically, the leader updates every element (i, j) of $\mathcal{L}_l$ using

$$L_l(i,j) = \sum_{m \in ROS_l} b_m(i,j), \;\; i, j \in \{1, \cdots, G_l\},$$

where

$$b_m(i,j) = \begin{cases} +1 & \text{if } A_m(t) = 1 \text{ AND } \mathcal{M}_l^{-1}(i,j) \in \overline{ROC}_m \\ -1 & \text{if } A_m(t) = 0 \text{ AND } \mathcal{M}_l^{-1}(i,j) \in \overline{ROC}_m \\ 0 & \text{otherwise} \end{cases}$$

and $\overline{ROC}_m$ is the set of all grid cells that are covered by the ROC of sensor m. In other words, the leader in a *distributed* fashion *Subtracts on Negative and Adds on Positive (dSNAP)*. The maximum of the scoring matrix points to the estimated position of the event source which is taken to be the center of the corresponding cell of $\mathcal{G}_l$. Let $\mathcal{L}_l(i^*, j^*) = \max_{i,j} \mathcal{L}_l(i, j)$, then the estimated source position is the center of $\mathcal{M}_l^{-1}(i^*, j^*)$. If two or more elements have the same maximum value, the estimated position is the centroid of the corresponding cell centres. Figure 3.15b illustrates how dSNAP works in a simple scenario assuming a square ROC.

The complexity of the algorithm is linear with respect to the number of elements of the $\mathcal{L}_l$ matrix, or $O(G_l^2)$ which is significantly more efficient than the centralized approach where the number of messages is proportional to the total number of nodes in the field (not just the neighbours) and the search for the maximum is over a scoring matrix that covers the entire field. The steps used by the leader node for constructing the scoring matrix are outlined in Algorithm 5 at Appendix D.

Selection of the *ROS* Size

In order to estimate the event location, leader node l needs to collect information from sensor nodes located inside its neighbourhood (ROS_l). Since energy efficiency is a major consideration in wireless sensor networks and communication is the most expensive operation in terms of energy, we would like to find the smallest ROS_l that achieves the desired objectives in terms of estimation accuracy. This is given by the following Lemma.

Lemma 3. *Assuming the leader is correctly alarmed (i.e., it falls in the ROI of the source) then its ROS_l radius $R_S = 2R_C$.*

Proof. If the leader is correctly alarmed (i.e, it falls in the source *ROI*) then the maximum of the scoring matrix should occur in the elements that correspond to the *ROC* of the leader (ROC_l). By definition, ROC_l is a disc of radius R_C centred at the leader location. Since, we do not know the exact location of the source, ROS_l should include all sensor nodes relevant to the estimation problem. The *ROC* of a sensor that is within a distance $2R_C$ from the leader can directly influence the results of the scoring matrix so we need to have $R_S \geq 2R_C$.

On the other hand, the *ROC* of any sensor located further away than $2R_C$ has no overlap with ROC_l and therefore it cannot affect the computation of the element with the maximum value. This implies that $R_S \leq 2R_C$, which completes the proof. □

3.4.4 Simulation Results on Source Identification

For all subsequent experiments we use a square 100×100 sensor field with $N = 625$ randomly deployed nodes, unless otherwise stated in the experiment, where the sensor readings are given by equation (3.26) with $c_k = 5000, \ \forall k$. Furthermore, we assume $w_n(t)$ to be white Gaussian noise $\mathcal{N}(0, 1)$. For the distributed localization algorithm dSNAP, we use grid resolution $g = 1$ and *ROC* radius $R_C = 10$. Finally, the mean values reported are the average over 100 Monte-Carlo simulations, each with a randomly deployed sensor field.

Two Sources Identification

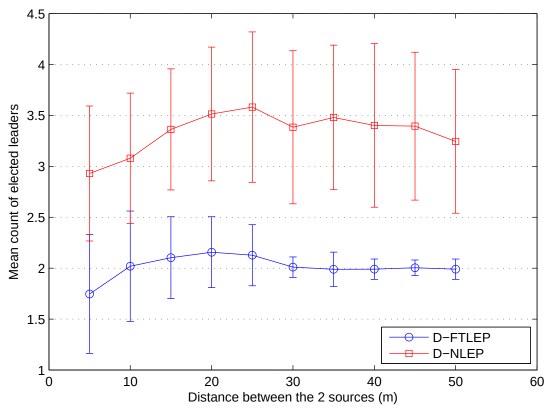
We evaluate the performance of D-FTLEP using a simulation scenario with two sources located in the middle of the sensor field as we vary the separation distance r_{sep} between them so that $r_{sep} \in [5\text{ m}, 50\text{ m}]$. The identification algorithm in D-FTLEP should correctly estimate the number of sources in the field by electing a leader for each source present. We are particularly interested in exploring the limits of the proposed algorithm as the two sources move closer together. The performance of D-FTLEP is compared against another "naive" leader election protocol referred to as D-NLEP, which allows *any* alarmed sensor node to become a leader irrespective of the value of F_n, whereas for D-FTLEP only alarmed sensor nodes with $F_n > 0$ are eligible for becoming leaders. Other than this, D-FTLEP and D-NLEP follow exactly the same protocol for estimating the number of targets in the field.

First, we investigate the performance of the identification algorithms in terms of the mean number of leaders elected, as we vary the separation distance r_{sep} between the 2 sources; see Figure 3.17a, where the error bars indicate the standard deviation. For the simulations we used 100 experiments and 10 repetitions for each value of r_{sep} producing 1000 results for each tested scenario. From these plots it becomes evident that D-FTLEP has very good performance and on average it correctly counts the sources, even in situations where the two sources are located almost next to each other ($r_{sep} = 5$ m). The reason lies in the distributed implementation. Even though the closely spaced sources are hard to distinguish, D-FTLEP can still separate the two sources because the superposition of the corresponding signals becomes strong enough, so it can be detected outside the ROS of the leader sensor. Thus, a second leader is also correctly elected. Our findings also indicate that the mean distance between the elected leaders and the associated source locations drops, as r_{sep} is increased. This confirms that the elected leaders are located very close to the sources.

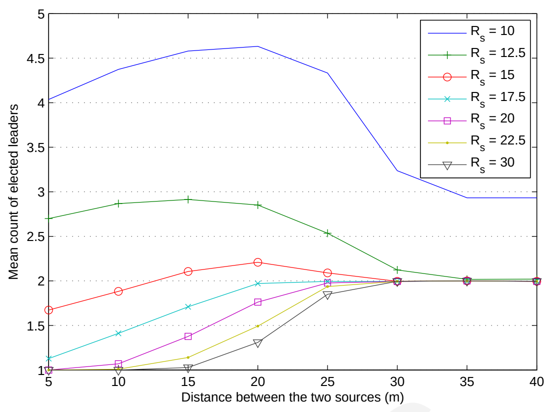
Second, we investigate the performance of D-FTLEP for various ROS sizes, i.e. by varying the radius R_S. Recall that R_S defines the neighbourhood where a sensor node can subscribe for information for calculating the F_n value according to D-FTLEP. Moreover, ROS is the neighbourhood where a leader node would send the suppression message in order to avoid the election of other leaders. So far, we have been using $R_S = 1.5R_C$. Since, ROS plays such an important role for the algorithm, in the following set of experiments we investigate its performance as we vary R_S for the two sources identification scenario. The simulation results for leader count for different values of r_{sep} are shown in Figure 3.17b as we vary R_S from 10 m to 30 m. For these experiments we kept $R_C = 10$ m. From the plot it becomes evident that on average $R_S = 15$ m (i.e., $1.5R_C$) achieves the most accurate results with a leader count close to two for all values of r_{sep} tested. Based on these results, for all subsequent experiments we continue to use $R_S = 1.5R_C$.

Scalability for Multiple Sources and Varying Node Density

In this section, we address scalability issues in the more general case where we have K randomly deployed sources in the field and evaluate the performance of the two identification algorithms (D-FTLEP and D-NLEP) by varying the percentage of faulty sensor nodes (suffering temporary mixed faults) and the node density in the field. In the simulation results that follow, the D-FTLEP algorithm is portrayed with solid lines, while the D-NLEP algorithm is shown with dotted lines.



(a) Number of elected leaders.

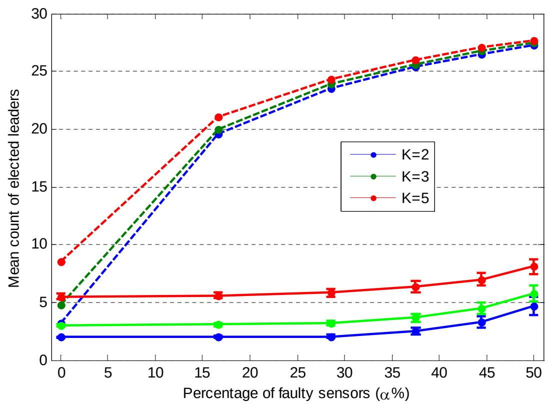


(b) Number of elected leaders for varying R_S.

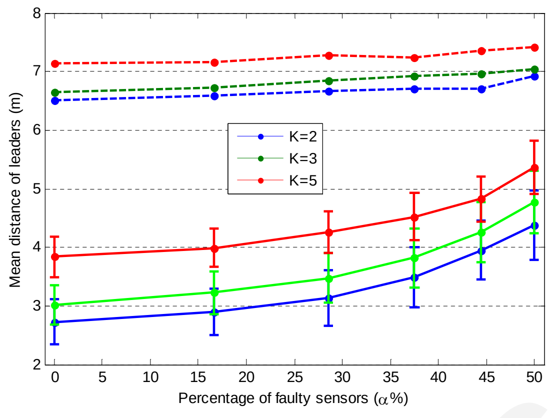
Figure 3.17: Performance of D-FTLEP and D-NLEP for two sources identification.

The performance of the identification algorithms in terms of the leader count and the leader distance from the actual source locations is depicted in Figure 3.18 for $K \in \{2, 3, 5\}$ sources as we vary α. In Figure 3.18a we observe that in the fault-free case ($\alpha = 0\%$) the D-FTLEP algorithm accurately counts the number of sources when $K = 2, 3$, while for $K = 5$ the leader count is 5.8. This is expected because by increasing the number of randomly deployed sources there is a higher probability that two or more sources are located very close to each other and this may lead to additional leaders becoming elected; see the two source simulation scenario in Figure 3.17a. Interestingly, the D-FTLEP algorithm is able to retain its accuracy for *all* different numbers of sources by electing the same number of leaders as in the fault-free case. In fact, the fault tolerance of D-FTLEP is evident even when $\alpha = 40\%$ of the sensor nodes give erroneous observations. At the same time, the D-NLEP identification algorithm (dotted lines) fails immediately when faults occur and ends up electing a large number of leaders in all cases. For instance, the number of elected leaders using D-NLEP is close to 20 for $K = 2$ source and $\alpha = 15\%$. To make things worse, the D-NLEP algorithm always elects leaders that are far (more than 6 m) from the actual source locations even in the absence of faults. In contrast, the D-FTLEP algorithm tends to elect leaders that are much closer to the sources, while the leader distance is slightly affected when the number of faulty sensors is increased.

Figure 3.19 depicts the results on the leader count and leader distance using $K = 5$ sources as we increase the number of sensor nodes in the field, while $\alpha \in \{0\%, 29\%, 44\%\}$. Looking at Figure 3.19a it is obvious that D-FTLEP is robust with respect to the number of sensor nodes for different percentages of faulty nodes and the number of elected leaders is always close to 5. On the other hand, the D-NLEP algorithm counts the number of sources correctly only in the fault-free case, while introducing faults affects its performance severely. Regarding the



(a) Leader count.

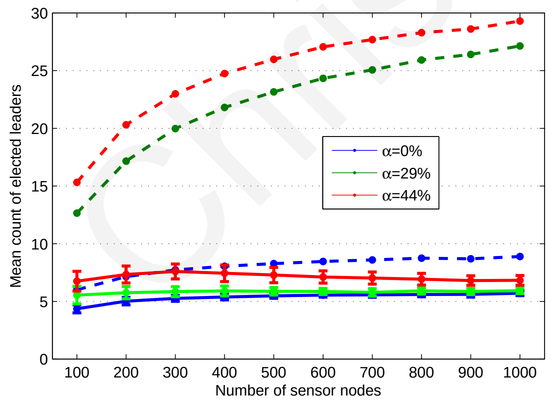


(b) Leader distance.

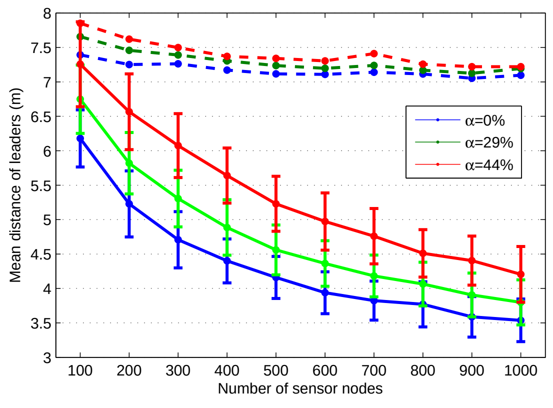
Figure 3.18: Performance of D-FTLEP and D-NLEP for multiple sources identification under temporary mixed faults.

distance between the elected leaders and the actual source locations, D-FTLEP tends to elect leaders that are on average further away when the number of sensor nodes in reduced. The leaders elected using the D-NLEP algorithm are always further away (more than 7 m) from the actual source locations regardless of the sensor node density, as shown in Figure 3.19b.

The above findings suggest that the distributed counterpart of the proposed ftTRACK architecture can easily scale and performs well for increasing number of moving sources in the field, as well as varying sensor node density.



(a) Leader count.



(b) Leader distance.

Figure 3.19: Performance of D-FTLEP and D-NLEP for $K = 5$ sources and varying sensor density under temporary mixed faults.

3.5 Chapter Summary

In this chapter, we study target tracking using a binary WSN and focus on improving the resilience to faults that disturb the original sensor observations. In this way, we manage to maintain high tracking accuracy, even when a large number of faulty sensors exist.

We introduce a new MC fault model to generate temporally correlated faults, which may occur unintentionally or due to a malicious attack. Using this fault model we are able to simulate different realistic types of binary sensor faults, including reverse status and stuck-at faults, that can be either transient or permanent. In addition, we can simulate spatially correlated faults.

Subsequently, we investigate the sensor health estimation problem because accurate and reliable sensor state estimates (i.e., either healthy or faulty) can greatly improve the fault tolerance of localization and tracking algorithms. Typically, such algorithms use all sensor contributions to determine the target location, while faulty sensors should not be considered because they may lead to accuracy degradation. To this end, we formulate the sensor health state estimation problem as a HMM and present three state estimators.

Next, we combine our sensor state estimators with localization and location smoothing algorithms and integrate them into a closed loop tracking system. The resulting ftTRACK architecture is a centralized solution that exploits spatiotemporal information for intelligently choosing only healthy sensors for estimating the unknown target location and tracking it, while it traverses the sensor field. Our simulation results indicate that ftTRACK can greatly improve the fault tolerance of the underlying target localization and tracking algorithms.

Finally, we work towards a distributed ftTRACK architecture to enable multiple target tracking, while addressing the limitations of the centralized ftTRACK architecture. To this end, we present a distributed fault tolerant leader election protocol for detecting and counting the number of targets in the field. We also describe a distributed version of the SNAP localization algorithm that can be run by each leader to localize the associated target. As part of our future work we will investigate the sensor health state estimation problem in the distributed environment, so that the elected leaders will employ the observations only from those neighbouring nodes considered to be healthy.

Chapter 4

Fault Tolerant Localization in WLAN

4.1 Background

4.1.1 Problem Formulation & Definitions

Traditionally, RSS fingerprinting consists of two phases, namely the *offline* (training) and the *online* (localization) phases.

In the *offline* phase, we consider a set of predefined reference locations $\{L : \ell_i = (x_i, y_i),\ i = 1, \ldots, l\}$ on a grid that spans the area of interest to collect RSS values from n APs. A reference fingerprint $\mathbf{r}_i = [r_{i1}, \ldots, r_{in}]^T$ associated with location ℓ_i, is a vector of RSS samples and r_{ij} denotes the RSS value related to the j-th AP. Usually, $\mathbf{r}_i$ is averaged over multiple fingerprints collected at ℓ_i to alleviate the effect of noise in RSS measurements and outlier values.

In the *online* phase, we exploit the reference data to obtain a location estimate $\widehat{\ell}$, given a new fingerprint $\mathbf{s} = [s_1, \ldots, s_n]^T$ measured at the unknown user location ℓ, using any desired fingerprinting algorithm as discussed in Section 2.2.1.

We assume that the reference RSS data collected in the offline phase are not corrupted. This assumption is not restricting because reference data can be validated using security and attack prevention or detection mechanisms [18] prior to deploying the localization system. Thus, we focus on non-cryptographic RSS-based attacks and failures that may occur during the online phase. Before describing the fault models we define the Region of Coverage (RoC) of an AP.

Definition 6. *The Region of Coverage denoted $RoC_j \subseteq L,\ j = 1, \ldots, n$ is the subset of reference locations inside the area of interest that contains those locations where the j-th AP is sensed during the offline phase.*

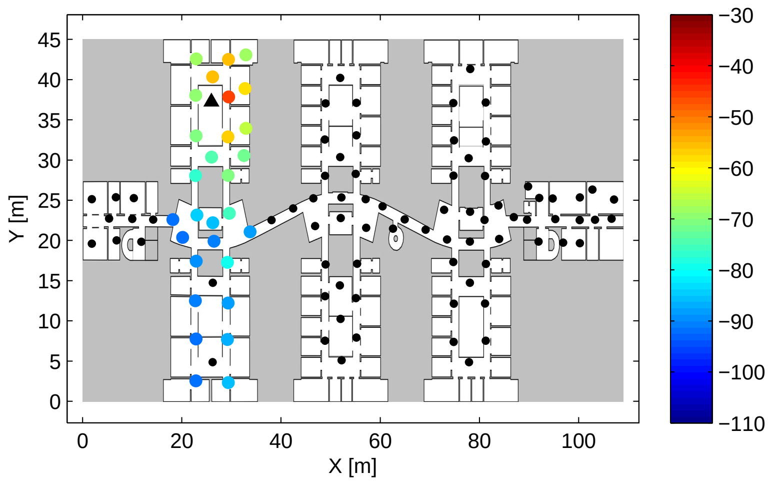


Figure 4.1: Example *RoC* of an AP in the *VTT dataset*.

For instance, all reference locations (small black dots) and the locations inside the *RoC* of a single AP (larger dots) in the *VTT dataset* are depicted in Figure 4.1. The AP is located in the top left wing (black triangle) and the colorbar indicates the mean RSS level from that AP at each location. The VTT experimental setup and corresponding dataset is described in Section 4.1.3.

4.1.2 Basic Fingerprinting Algorithms

For easy reference we briefly outline two basic fingerprinting algorithms that will be later used for enabling the detection of faulty APs and for building fault tolerant counterparts.

Deterministic approach

The NN algorithm [9, 10], is a deterministic approach that estimates location by minimizing the Euclidean distance d_i, between the observed fingerprint during localization $\mathbf{s}$ and the reference fingerprints $\mathbf{r}_i$ in the radiomap

$$\widehat{\ell}(\mathbf{s}) = \arg\min_{\ell_i} d_i, \;\; d_i = \sqrt{\sum_{j=1}^{n}\left(r_{ij} - s_j\right)^2}. \tag{4.1}$$

Essentially, all reference locations are ordered according to increasing d_i and location ℓ_i with the shortest distance between $\mathbf{r}_i$ and $\mathbf{s}$ in the n-dimensional RSS space is returned as the location estimate. A variant known as K-Nearest Neighbour (KNN) method estimates the unknown location at the centroid of the corresponding K ordered candidate locations.

Probabilistic approach

In the probabilistic approach, location ℓ is treated as a random vector that can be estimated by calculating the conditional probabilities $p(\ell_i|\mathbf{s}),\ i = 1, \ldots, l$ (*posterior*) given the observed fingerprint $\mathbf{s}$. We adopt the approach of [125] in order to calculate the expected value of ℓ, i.e., $\widehat{\ell} = \mathrm{E}[\ell|\mathbf{s}] = \sum_{i=1}^{l} \ell_i p(\ell_i|\mathbf{s})$, which provides the Minimum Mean Square Error (MMSE) location estimate. Applying Bayes' rule we get

$$p(\ell_i|\mathbf{s}) = \frac{p(\mathbf{s}|\ell_i)p(\ell_i)}{p(\mathbf{s})} = \frac{p(\mathbf{s}|\ell_i)p(\ell_i)}{\sum_{i=1}^{l} p(\mathbf{s}|\ell_i)p(\ell_i)}, \tag{4.2}$$

where $p(\mathbf{s}|\ell_i)$ is the *likelihood* of $\mathbf{s}$ given ℓ_i, $p(\ell_i)$ is the *prior* probability of ℓ_i and $p(\mathbf{s})$ is a normalizing constant. The *prior* can be assumed to be uniformly distributed, i.e., $p(\ell_i) = 1/l,\ \forall \ell_i$, thus the problem reduces to estimating $p(\mathbf{s}|\ell_i)$. Assuming that the RSS measurements from APs are independent we get

$$p(\mathbf{s}|\ell_i) = \prod_{j=1}^{n} p(s_j|\ell_i). \tag{4.3}$$

We estimate the probability $p(s_j|\ell_i)$ of observing the RSS value s_j from the j-th AP at location ℓ_i during localization by using Gaussian kernels centred at r_{ij} with user-defined width σ.

4.1.3 Experimental Setups & Datasets

In the following we describe the localization areas and measurement setups that we have used to collect experimental RSS data. Our datasets contain reference data, collected at predefined reference locations with one WLAN-enabled mobile devices, which are used to build the required radiomap for the fingerprinting algorithms. Moreover, the datasets contain additional test data, collected at several test locations with the same mobile device, which are used for the validation of the developed algorithms.

VTT Dataset

For this dataset, we collected the reference data in a typical modern office environment on the second floor of a three storey building at VTT Technical Research Centre, Finland[1]. The size of the floor is around 5.000 m^2 and it consists of eight wings containing offices and meeting rooms connected with corridors. There are 31 Cisco Aironet APs installed throughout the

[1]`http://www.vtt.fi/`

building that use the IEEE 802.11b/g standard, while 10 of them are uniformly deployed on the second floor. We used a Fujitsu-Siemens Pocket Loox smart phone with Windows Mobile operating system to collect RSS measurements from all APs at 107 distinct reference locations on the second floor. These locations are separated by 2-3 meters and form a grid that covers all public places and meeting rooms.

A total of 3210 reference fingerprints, corresponding to 30 fingerprints per reference location, were collected at the rate of 1 sample per second. Due to the open plan interior design, the APs can be partially sensed on the second floor, and the average number is 9.7 APs per reference location. The floorplan of the experimentation area and the reference locations are depicted in Figure 4.2a, while the colorbar indicates the number of APs sensed at each location. The RSS values range from -101 dBm to -34 dBm.

For testing purposes, we collected additional fingerprints on the second floor by walking at a constant speed over a path that consists of 192 locations. One fingerprint is recorded at each location, and the same path is sampled 3 times for a total of 576 test fingerprints.

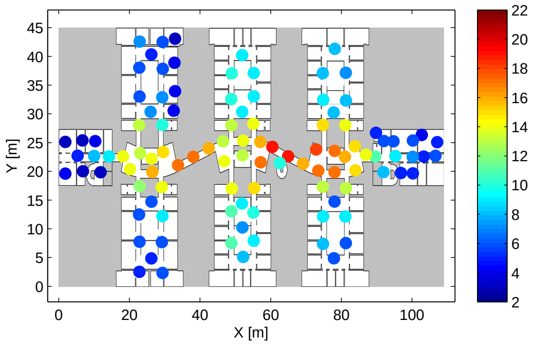
KIOS-A Dataset

We collected our RSS data inside the premises of KIOS Research Center, University of Cyprus[2]. The total area is around 560 m^2 and the floor consists of several open cubicle-style and private offices, labs, a conference room and corridors. We have installed 9 local WLAN APs (IEEE 802.11b/g) that provide full coverage throughout the floor. In addition, there are 64 other neighbouring APs that can be partially sensed in different sections.

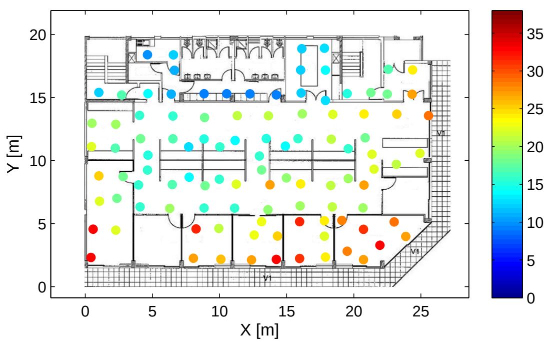
We used a HP iPAQ hw6915 PDA with Windows Mobile operating system to collect RSS measurements from all 73 APs at 105 distinct reference locations. These locations are separated by 2-3 meters and form a grid that covers all public places. A total of 4200 training fingerprints were collected for our radiomap, corresponding to 40 fingerprints per location. The floorplan of the experimentation area and the reference locations are depicted in Figure 4.2b, where the color bar indicates the number of APs sensed at each location. On average approximately 20 APs can be sensed per location, while the raw RSS values range from -90 dBm to -10 dBm.

For testing purposes, we have also collected 1920 fingerprints at 96 locations (20 fingerprints per location) inside the experimentation area, while most of the test locations do not coincide with the reference locations.

[2]`http://www.kios.ucy.ac.cy/`



(a) VTT experimental setup.



(b) KIOS experimental setup.

Figure 4.2: Reference locations and number of sensed APs in the setups.

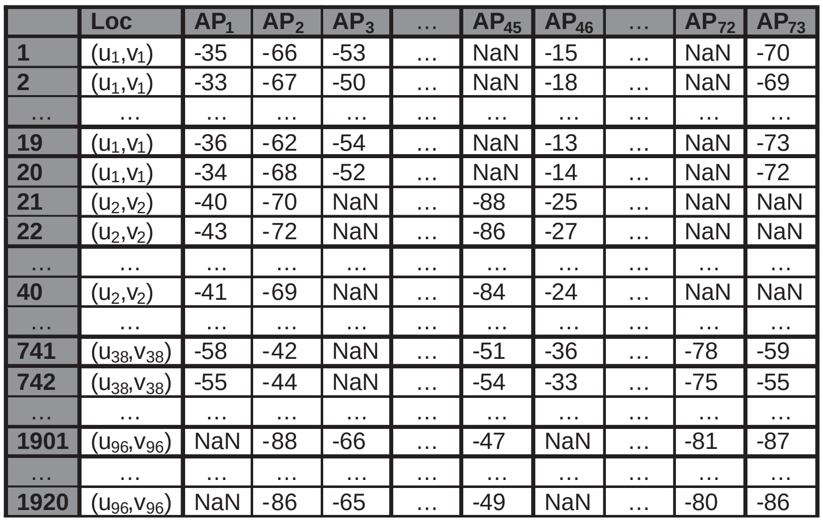
4.2 Fault Models

In the following we introduce several fault models to capture the effect of AP malfunctions or malicious attacks during localization [74]. Then, we describe how these new models can be simulated using the original test data to allow extensive evaluation and comparison of fingerprinting algorithms. Each model is followed by a short discussion on the feasibility of the underlying attack or the occurrence probability of the relevant failure, in both *private* and *public* WLAN-based localization systems. As discussed in Section 1.2, the former rely only on the APs deployed inside a private fully-controlled network, while the latter consider all available APs, including neighbouring public and residential APs.

4.2.1 AP Failure Model

In this scenario we consider the case where several APs used in the offline phase are not available during localization. This type of fault may occur due to power outages, WLAN system maintenance, AP firmware upgrades, etc., or can be caused by unpredicted AP failures or malfunctions. Regarding *public* localization systems, an AP listed in the database may be temporarily shut down or permanently removed by its owner. This constitutes an AP failure during localization from the user perspective, until the database is updated. Alternatively, when an attack is assumed an adversary can easily cut off the power supply of some APs or use specialized equipment to severely jam the communication channels and make the attacked APs unavailable in order to compromise the localization performance. Jamming attacks can be easily launched, as described in [140].

We simulate the *AP Failure* model by removing the valid RSS values of randomly selected APs in the original test fingerprints. For instance, if AP_2 is faulty we replace all RSS values in



	Loc	AP_1	AP_2	AP_3	…	AP_{45}	AP_{46}	…	AP_{72}	AP_{73}
1	(u_1,v_1)	-35	-66	-53	…	NaN	-15	…	NaN	-70
2	(u_1,v_1)	-33	-67	-50	…	NaN	-18	…	NaN	-69
…	…	…	…	…	…	…	…	…	…	…
19	(u_1,v_1)	-36	-62	-54	…	NaN	-13	…	NaN	-73
20	(u_1,v_1)	-34	-68	-52	…	NaN	-14	…	NaN	-72
21	(u_2,v_2)	-40	-70	NaN	…	-88	-25	…	NaN	NaN
22	(u_2,v_2)	-43	-72	NaN	…	-86	-27	…	NaN	NaN
…	…	…	…	…	…	…	…	…	…	…
40	(u_2,v_2)	-41	-69	NaN	…	-84	-24	…	NaN	NaN
…	…	…	…	…	…	…	…	…	…	…
741	(u_{38},v_{38})	-58	-42	NaN	…	-51	-36	…	-78	-59
742	(u_{38},v_{38})	-55	-44	NaN	…	-54	-33	…	-75	-55
…	…	…	…	…	…	…	…	…	…	…
1901	(u_{96},v_{96})	NaN	-88	-66	…	-47	NaN	…	-81	-87
…	…	…	…	…	…	…	…	…	…	…
1920	(u_{96},v_{96})	NaN	-86	-65	…	-49	NaN	…	-80	-86

Figure 4.3: Sample RSS values in the test fingerprints of the *KIOS-A dataset*.

the corresponding column of our test data from the *KIOS-A dataset*, as shown in Figure 4.3, with *NaN* values. This designates that AP_2 is not available during localization.

4.2.2 AP Relocation Model

This scenario considers the effect of relocating a set of APs and thus a faulty AP is sensed during localization inside an area that can be different than the expected one. This may happen in case that the AP is moved to a new location, e.g., for network operation reasons, and the data in the radiomap are not updated by collecting additional fingerprints to cater for the affected areas. In *public* localization systems this can happen quite often because the AP owners may move them around inside their private properties.

On the other hand, an adversary may launch an attack with the same effect by physically relocating an AP. Alternatively, the attacker can impersonate a specific AP (i.e., Sybil attack), while at the same time eliminates the AP signals through jamming. AP impersonation can be easily implemented, especially in *public* localization systems, because rogue APs can forge the MAC addresses of legitimate APs and transmit at arbitrary power levels within their physical capabilities. Details on the feasibility of impersonation and replication attacks can be found in [137], where the application of these attacks on the *Skyhook public* localization system is reported.

We simulate the *AP Relocation* model by replacing the RSS readings of the corrupted AP in the test data with the values of another randomly selected AP. For example, if AP_3 is moved to a new location close to AP_{45}, then we replace all RSS values of AP_3 with the respective values of AP_{45}; see Figure 4.3.

4.2.3 False Negative Model

In this model, the assumption is that an AP may no longer be sensed in some locations inside its original *RoC*. This can happen accidentally if furniture or equipment is moved, so that the propagation path is blocked and the AP signal cannot be sensed in locations where it was previously weak. This type of fault may also occur in *public* localization systems due to the construction of a building in the vicinity of that AP.

We simulate this model by ignoring valid RSS readings for a set of APs in a number of test fingerprints. For instance, if AP_1 suffers from *False Negative* fault we replace with *NaN* value some of its RSS values in our original test data, e.g., corresponding to 70% of the test locations (randomly selected), where that AP was previously sensed. Note that the *AP Failure* model is an extreme case of this model.

4.2.4 False Positive Model

Another scenario is when an AP is sensed during localization in locations outside its original *RoC*. Contrary to the *False Negative* model, this can happen unintentionally in case a heavy object or equipment, which was previously obstructing the propagation path, was moved after collecting the reference data. Essentially, the transmission signals can travel further and make that AP hearable in locations outside its original *RoC*. An attack scenario that manifests in a similar manner is when a rogue AP is deployed and programmed to replicate an existing AP. In this fashion, the corrupted AP is thereafter sensed during localization in locations possibly far beyond its original *RoC*.

The *False Positive* model is simulated by injecting random RSS values to the test data for a set of APs that would not be sensed otherwise in those locations where the respective test fingerprints were collected. For example, if AP_{72} suffers from *False Positive* fault we replace with random RSS values some of its *NaN* values in our original test data, e.g., corresponding to 70% of the test locations (randomly selected), where that AP would not be sensed otherwise.

4.3 Fault Detection

Standard localization methods do not take faults into account and cannot guarantee acceptable performance in terms of accuracy. Thus, it is important to use a robust detection scheme, as a first step, to signify that the observed fingerprint $\mathbf{s}$ is corrupt due to faults.

A potential fault indicator is the number of missing APs during localization, denoted n_{miss}.

In the presence of faults a subset of the APs, which would otherwise be sensed under normal conditions, are no longer present. In this case, several missing RSS values are injected into the observed fingerprint, thus increasing the value of n_{miss}. Our experimental findings indicate that n_{miss} is not a robust fault indicator, because in most practical implementations the APs provide only partial coverage in the area of interest and there are several missing APs, even under normal conditions. Such indicator is suitable only for small scale setups where all APs cover the whole area; however, these cases are of limited interest in practice.

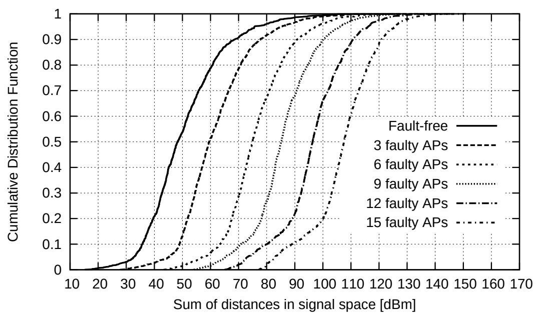
In the following we discuss some fault indicators inspired by RSS fingerprinting methods and describe how to select an appropriate threshold for detecting faults that follow the *AP Failure* and *AP Relocation* models. Some of these fault indicators were discussed in [31], however, they were only employed to detect signal attenuation or amplification attacks.

The main idea in all our fault detection schemes is the selection of an appropriate threshold γ based on the distribution of the relevant fault indicator when no faults are present, i.e., by using the fingerprints contained in the original test set. Specifically, we select γ by setting the acceptable false detections in the fault-free case. Under faults the value of the fault indicator is increased because the corrupt fingerprint is very dissimilar to the fingerprints in the radiomap and a fault can be detected during localization if the threshold γ is violated. We have found that selecting γ according to this empirical methodology provides a good compromise between low false detection rate, especially in the fault-free case, and high correct detection rate as the percentage of faulty APs is increased [85].

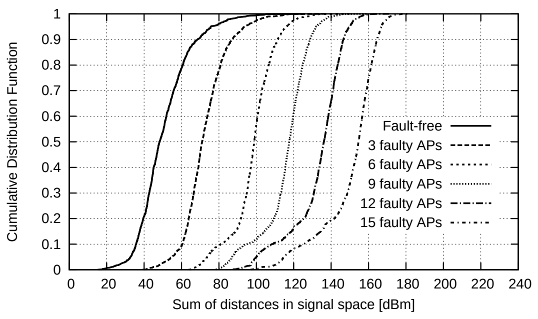
4.3.1 Distance-based Detection

In the context of the KNN method we can exploit the distances d_i that are already computed with equation (4.1) to decide whether fingerprint $\mathbf{s}$ is corrupt. We may use a fault indicator based on these distances, e.g., the distance from the K-th nearest neighbor, and the intuition is that under faults the value of the indicator will violate a certain threshold. For instance, authors in [31] use the distance from the nearest neighbour (i.e., $K = 1$), denoted D_s, to detect signal attenuation or amplification attacks. We use the sum of distances to the K nearest neighbours, denoted $D_{sum}^{(K)}$, as a fault indicator which was proved experimentally to be more robust compared to D_s, under the fault models described in Section 4.2. Note that in case $K = 1$ these two fault indicators are equivalent.

As a first step in our fault detection scheme, we select an appropriate threshold γ based on the distribution of $D_{sum}^{(K)}$ for the fingerprints contained in the original test set. Subsequently, a



(a) *AP Failure* model.



(b) *AP Relocation* model.

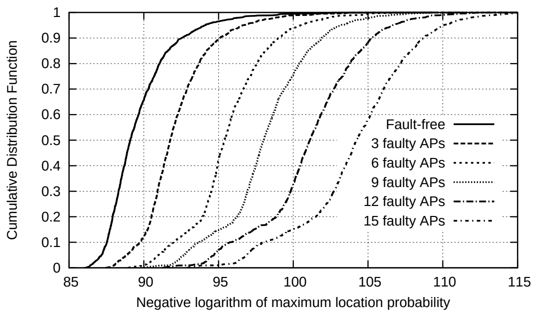
Figure 4.4: CDF of the $D_{sum}^{(2)}$ fault indicator under different fault models.

fault can be detected during localization if $D_{sum}^{(K)} > \gamma$ for the currently observed fingerprint.

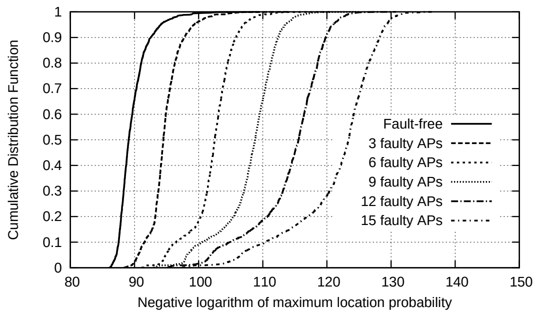
We examined several values for K and the best performance, in terms of fault detection accuracy, was obtained for $K = 2$. The CDF of $D_{sum}^{(2)}$ is plotted in Figure 4.4a for the fault-free case (solid line) and when we inject faults in the test fingerprints according to the *AP Failure* model. The first observation is that the CDF curves are shifted to the right as the number of faulty APs is increased. In particular, we observe that in the fault-free case $D_{sum}^{(2)}$ is below 76 dBm for 95% of the time and this corresponds to the 88th, 53th, 15th, 7th and 1st percentile as the number of faulty APs is increased from 3 to 15. Thus, setting $\gamma = 76$ dBm leads to 5% false detections in the fault-free case and suggests that the corrupt fingerprints will be detected with high probability.

Faults under the *AP Relocation* model seem to affect the RSS fingerprints more heavily and the CDF curves are sharper, as shown in Figure 4.4b. In this case our threshold value corresponds to the 66th, 7th and 0th percentile for 3, 6 and more than 6 faulty APs, respectively. This is significantly lower compared to the *AP Failure* model case, especially when few APs are faulty, and this is a strong indication that this type of fault will be detected more easily. From another perspective, less faulty APs will be required to detect a corrupt fingerprint under the *AP Relocation* model, compared to the *AP Failure* model.

When the *False Negative* model is considered, our threshold value corresponds to the 86th, 64th, 21st, 11th and 7th percentile, as the number of faulty APs is increased from 3 to 15. These percentiles are similar with the case of *AP Failure* faults indicating that corrupt fingerprints that follow the *False Negative* model will also be detected effectively. Regarding the *False Positive* model, our threshold value corresponds to the 0th percentile when only 3 APs are faulty, suggesting that this type of faults can heavily disturb the localization fingerprints. Thus, the corrupt fingerprints that follow the *False Positive* model can be easily detected.



(a) *AP Failure* model.



(b) *AP Relocation* model.

Figure 4.5: CDF of the P_{max} fault indicator under different fault models.

4.3.2 Likelihood-based Detection

Regarding the probabilistic MMSE method, we use the likelihoods $p(\mathbf{s}|\ell_i),\ i = 1, \ldots, l$ that are already calculated with equation (4.3) to detect possible faults. As discussed in [31], we build a fault indicator based on the maximum likelihood of the candidate locations ℓ_i, which is formally given by

$$P_{max} = -\log \max_{\ell_i} p(\mathbf{s}|\ell_i). \tag{4.4}$$

The CDF curves of P_{max} are plotted in Figure 4.5a in the presence of *AP Failure* faults. Similar behaviour to the $D_{sum}^{(2)}$ indicator is observed and in the fault-free case $P_{max} < 94$ for 95% of the time. Assuming that we can tolerate around 5% false detections, we set $\gamma = 94$. This corresponds to the 84th, 23th, 11th, 2nd and 1st percentile as the number of faulty APs is increased from 3 to 15. On the other hand, when the *AP Relocation* model is considered, $\gamma = 94$ corresponds to the 40th, 2nd and 0th percentile for 3, 6 and more than 6 faulty APs; see Figure 4.5b. Regarding the *False Negative* model, our threshold value corresponds to the 81st, 39th, 13th, 8th and 4th percentile, as the number of faulty APs is increased from 3 to 15. When the *False Positive* model is considered, the threshold corresponds to the 0th percentile when more that 3 APs are faulty.

To summarize, the P_{max} fault indicator has similar behaviour with the $D_{sum}^{(2)}$ distance-based indicator discussed above. Another possible fault indicator that can be applied to probabilistic fingerprinting algorithms is the negative logarithm of the sum of likelihoods, i.e., $P_{sum} = -\log \sum_{i=1}^{l} p(\mathbf{s}|\ell_i)$. This fault indicator demonstrates similar behaviour with P_{max} under all fault models, thus in the following only P_{max} is considered.

4.3.3 Performance Evaluation

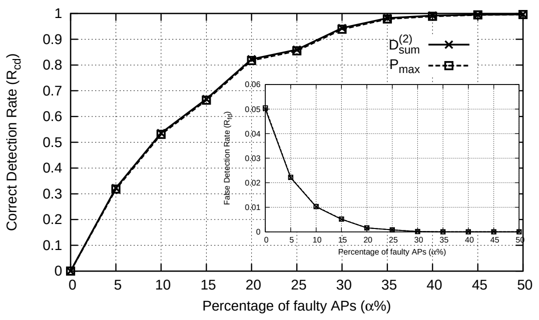
We evaluate the fault indicators by adopting two performance metrics, namely the correct detections rate (R_{cd}) and the false detections rate (R_{fd}) that are defined as the ratio of the test fingerprints detected to be corrupt, either correctly or falsely, over all test fingerprints. In our previous work, we investigated the detection rates R_{cd} and R_{fd} of the $D^{(2)}_{sum}$ indicator under *AP Failure* faults for various threshold values and concluded that based on the selection of γ there is a trade off between the corrupt fingerprints that will go undetected when faults are present and false detections in the fault free-case [85].

For our evaluation we apply the *AP failure* and *AP Relocation* fault models to corrupt the original test data and the results are averaged over 100 runs using randomly selected subsets of faulty APs in each run[3]. We keep the thresholds fixed for $D^{(2)}_{sum}$ and P_{max}, i.e., the γ values are selected as discussed in Sections 4.3.1 and 4.3.2, and compare both fault indicators in terms of fault detection. The detection rate curves under *AP Failure* faults are plotted as a function of the percentage of faulty APs (α%) in Figure 4.6a. We observe that the performance of $D^{(2)}_{sum}$ and P_{max} is almost identical and the R_{cd} grows beyond 0.85 when $\alpha > 25\%$. When the percentage of faulty APs is low, i.e. $\alpha \leq 10\%$, the R_{cd} for $D^{(2)}_{sum}$ and P_{max} does not exceed 0.6. In this case it is hard to discern whether the missing values in the fingerprints are due to APs that have failed, because the APs do not provide ubiquitous coverage and there are missing values at different locations even under normal conditions. Thus, some corrupt fingerprints can be undetected. However, as it will be shown later, when few APs are faulty then the localization accuracy is not severely degraded, thus failing to detect these faults is not crucial. Regarding false detections, the R_{fd} for both indicators is less than 0.01 when $\alpha \geq 15\%$; see the inset plot in Figure 4.6a.

When the *AP Relocation* model is considered the $D^{(2)}_{sum}$ and P_{max} indicators are very effective in detecting this type of fault. For instance, the R_{cd} is close to 0.9 for both indicators when $\alpha = 15\%$, as shown in Figure 4.6b, compared to 0.67 in the case of *AP Failure* faults. Moreover, false detections decrease rapidly and the R_{fd} drops below 0.01 when just 5% of the APs are faulty; see the inset plot in Figure 4.6b. As pointed out previously, *AP Relocation* faults corrupt the RSS fingerprints severely leading to higher R_{cd} and lower R_{fd} with only few faulty APs.

These results verify that both $D^{(2)}_{sum}$ and P_{max} are robust fault indicators that detect the

[3]Results on the detection capability under the *False Negative* and *False Positive* models are not reported for brevity because they are similar with our findings using the *AP failure* and *AP Relocation* models, respectively.


(a) Detection rates under the *AP Failure* model.

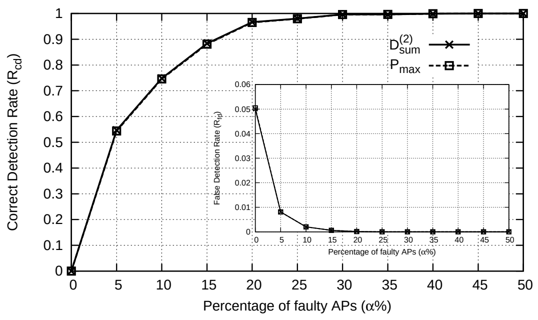

(b) Detection rates under the *AP Relocation* model.

Figure 4.6: Correct and false detection rates for different fault indicators.

corrupt fingerprints, which would otherwise degrade the localization accuracy. More importantly, in an attack scenario an adversary will not be able to compromise the localization system without being detected. Moreover, selecting the threshold for each fault indicator with the empirical methodology discussed previously delivers a high level of correct detections, especially when the number of faulty APs is increased.

Notably, our fault detection scheme, based either on the $D_{sum}^{(2)}$ or P_{max} fault indicators, can be applied in any fingerprinting system to signify the presence of faults, regardless of the underlying localization algorithm. However, distinguishing between different types of faults is not possible with either approach. Fault identification is out of the scope of this thesis and is part of our ongoing research.

4.4 Fault Tolerant Localization

The standard KNN and MMSE localization algorithms cannot provide the required level of accuracy when faults are present in the localization system [74]. Another KNN variant, referred to as MED, uses a median-based distance metric to improve fault tolerance in case of failures or incorrect RSS readings [95]. In this case, location is estimated by

$$\widehat{\ell}(\mathbf{s}) = \arg\min_{\ell_i} d_i, \;\; d_i = \operatorname{med}_{j=1}^{n}\left(r_{ij} - s_j\right)^2. \tag{4.5}$$

Our approach is to utilize the fault detection schemes described previously and modify the localization algorithms in order to build hybrid KNN and MMSE counterparts that achieve higher resilience to faults.

4.4.1 Localization Under AP Failure Faults

Hybrid KNN Algorithm

The distances d_i computed with equation (4.1) can be viewed as

$$d_i = \sqrt{\sum_{j\in R_i\cap S} d_{ij} + \sum_{j\in R_i\setminus S} d_{ij} + \sum_{j\in S\setminus R_i} d_{ij}}, \tag{4.6}$$

where $d_{ij} = \left(r_{ij} - s_j\right)^2$, while R_i and S denote the sets of APs that are present in fingerprints $\mathbf{r}_i$ and $\mathbf{s}$, respectively. The first term refers to the intersection of R_i and S and represents the distance with respect to those APs that are common in $\mathbf{r}_i$ and $\mathbf{s}$. The second term employs those APs that are sensed in $\mathbf{r}_i$ and not in $\mathbf{s}$, while the last term considers those APs that are found in $\mathbf{s}$ and not in $\mathbf{r}_i$.

In practical deployments a small constant is used to replace the missing RSS values s_j and r_{ij} in equation (4.6). This is effective in the fault-free case, because all APs that are not found in common between $\mathbf{r}_i$ and $\mathbf{s}$ are penalized. However, in the presence of faults, the distances d_i can be affected and this leads to the wrong ordering of candidate locations, thus degrading the localization accuracy. In order to mitigate errors introduced by faulty APs, we employ the following distance metric given by

$$d_i' = \sqrt{\sum_{j\in R_i\cap S} d_{ij} + \sum_{j\in S\setminus R_i} d_{ij}}. \tag{4.7}$$

This metric ignores faulty APs in $R_i\setminus S$ and was shown to improve the fault tolerance of the standard KNN method, especially when a large number of APs are faulty [74,85]. We also expect this metric to be equally effective in the case of faults that follow the *False Negative* model. Our hybrid KNN algorithm, denoted as H-KNN, incorporates the detection mechanism described in Section 4.3.1 and couples that with the modified distance metric given in equation (4.7) to provide smooth performance degradation in the presence of *AP Failure* faults. The required threshold γ can be selected using the methodology described in Section 4.3.1. The steps of the H-KNN algorithm are listed in Algorithm 6 at Appendix E.

Hybrid MMSE Algorithm

We follow the same approach for the probabilistic MMSE method and by using the notation introduced previously the likelihood $p(\mathbf{s}|\ell_i)$ in equation (4.3) can be viewed as

$$p(\mathbf{s}|\ell_i) = \prod_{j\in R_i\cap S} p(s_j|\ell_i) \prod_{j\in R_i\setminus S} p(s_j|\ell_i) \prod_{j\in S\setminus R_i} p(s_j|\ell_i). \tag{4.8}$$

Missing RSS values in fingerprint $\mathbf{s}$ due to faults can severely affect the calculation of $p(\mathbf{s}|\ell_i)$ because these values are replaced with a small constant. We propose to compute the likelihood according to

$$p'(\mathbf{s}|\ell_i) = \prod_{j \in R_i \cap S} p(s_j|\ell_i) \prod_{j \in S \setminus R_i} p(s_j|\ell_i). \tag{4.9}$$

Essentially, the faulty APs that fall in the subset $R_i \setminus S$ are ignored and the fault tolerance of the MMSE method can be greatly improved. This also holds for faults under the *False Negative* model. Algorithm 7 in Appendix E provides the details of the proposed hybrid MMSE algorithm, denoted H-MMSE, that combines the detection mechanism based on the P_{max} fault indicator with the fault tolerant likelihood calculation formulated in equation (4.9). The required threshold γ can be selected using the methodology described in Section 4.3.2.

4.4.2 Localization Under AP Relocation Faults

The modified distance metric in equation (4.7), which is employed in the proposed H-KNN algorithm, greatly improves fault tolerance under the *AP Failure* model. However, our preliminary results indicated that it is not very effective under the *AP Relocation* and *False Positive* models, contrary to the median-based metric, that improved to some extent the performance of the standard KNN method [74]. Thus, the median-based metric is a good candidate for mitigating *AP Relocation* and *False Positive* faults. On the other hand, the median-based metric employed in the MED method performs poorly in the fault-free case or when half of the APs are faulty [69, 85].

To this end, we develop the hybrid MED algorithm, denoted H-MED, that combines the Euclidean distance metric in equation (4.6) with the median-based metric in equation (4.5) and switches to the latter in case of fault detection, as detailed in Algorithm 8 in Appendix E.

4.4.3 Experimental Results

We consider the *KIOS-A* experimental setup, described in Section 4.1.3, and investigate fault tolerance with respect to the accuracy degradation when faults occur. In particular, an algorithm is considered as fault tolerant, if the mean positioning error $\mathcal{E}$ does not increase rapidly as the percentage of faulty APs $\alpha\%$ is increased. From another perspective, we may select an acceptable upper bound on the performance, e.g., $\mathcal{E}_{ub} = 5$ m, and examine the percentage of faulty APs that each algorithm can tolerate. We apply the fault models described in Section 4.2 to corrupt the original test data and the results for $\mathcal{E}$ are averaged over 100 runs using randomly selected subsets of faulty APs in each run.

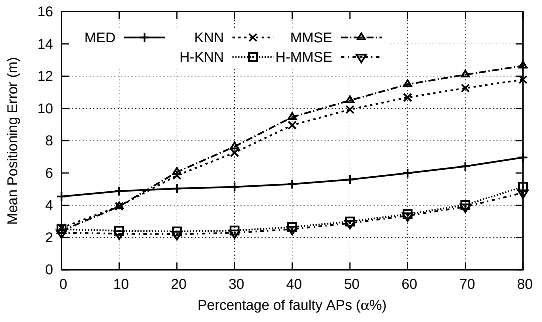
Performance Under *AP Failure* Faults

In the fault-free case, $\mathcal{E}$ is around 2.4 m for the standard KNN and MMSE methods, as well as for the hybrid H-KNN and H-MMSE algorithms; see Figure 4.7a for $\alpha = 0\%$. On the other hand, the MED algorithm performs poorly and the mean error is 4.55 m. As α is increased, $\mathcal{E}$ grows rapidly for KNN and MMSE methods and when $\alpha \geq 20\%$ they both fail to provide acceptable performance. In contrast, both hybrid algorithms exhibit smoother performance degradation. For instance, in an extreme scenario where half of the APs are faulty ($\alpha = 50\%$), $\mathcal{E} = 3$ m for H-KNN compared to $\mathcal{E} = 9.93$ m for KNN, while the standard deviation (std) of the localization error is 2.6 m and 5.6 m, respectively. Similar behaviour is observed when we compare the H-MMSE and MMSE algorithms. The MED method also seems to be more resilient to faults than KNN and MMSE; when $\alpha = 50\%$, $\mathcal{E}$ is increased by 1 m with respect to the fault-free case, however $\mathcal{E} = 5.6$ m is still much higher compared to our hybrid algorithms. Moreover, if $\mathcal{E}_{ub} = 5$ m is an acceptable upper bound, then both hybrid algorithms can tolerate up to 80% faulty APs compared to 30% for MED and only 10% for KNN and MMSE algorithms.

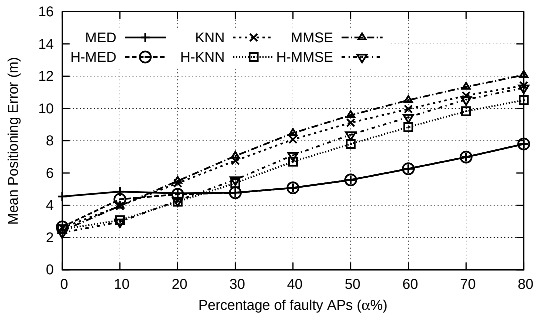
Performance Under *AP Relocation* Faults

The performance of the localization methods in the presence of *AP Relocation* faults is depicted in Figure 4.7b. When the percentage of relocated APs remains below 20%, then the H-MED algorithm exhibits lower $\mathcal{E}$ compared to the original MED method. Beyond that point H-MED is identical to the MED method, because the corrupt fingerprints are detected with high probability ($R_{cd} = 0.97$, as shown in Figure 4.6b) and the H-MED algorithm always switches to the median-based metric. In this fashion low localization error is maintained in the fault-free case, while the performance under *AP Relocation* faults is better than the other methods.

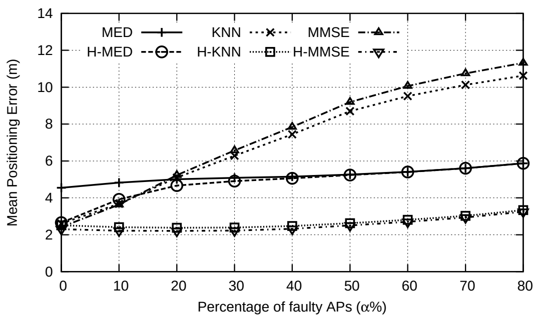
In the case that $\alpha \geq 30\%$ the KNN, MMSE, H-KNN and H-MMSE are all less fault tolerant compared to H-MED. For instance, when half of the APs are relocated $\mathcal{E} = 5.57$ m for H-MED, compared to other algorithms for which $\mathcal{E} > 7.8$ m. Moreover, in case $\mathcal{E}_{ub} = 5$ m is acceptable, H-MED can tolerate up to 40% of faulty APs compared to 20% for H-KNN or H-MMSE and 10% for KNN and MMSE methods. Note that the H-MED algorithm is fault tolerant, however *AP Relocation* faults are more difficult to mitigate in comparison with *AP*



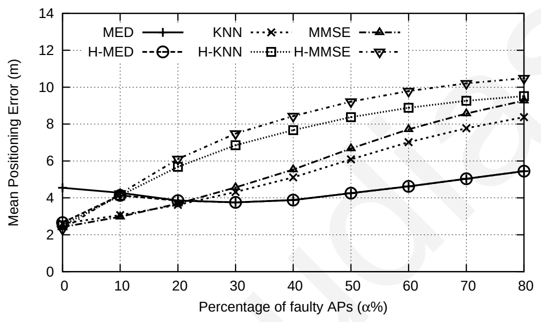
(a) *AP Failure* model.



(b) *AP Relocation* model.



(c) *False Negative* model.



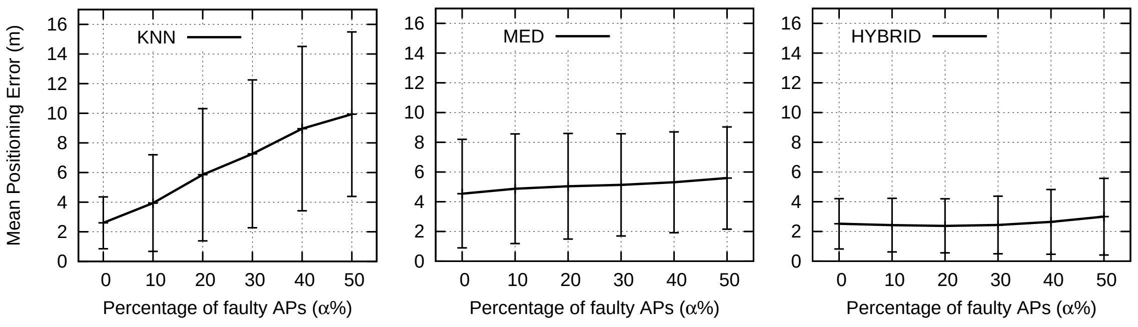
(d) *False Positive* model.

Figure 4.7: Fault tolerance of fingerprinting algorithms under various types of faults.

Failure faults, where the proposed hybrid algorithms can tolerate up to 80% of faulty APs. Moreover, as already discussed in Section 4.3.3, the $D_{sum}^{(2)}$ and P_{max} indicators cannot distinguish between the two different types of faults. Therefore, new mechanisms are required in the future in order to identify faults and select the most appropriate algorithm.

Performance Under *False Negative* and *False Positive* Faults

When the fingerprints are corrupted using the *False Negative* model, the behaviour of the hybrid algorithms is similar with the case of the *AP Relocation* model, as shown in Figure 4.7a and Figure 4.7c, and both H-KNN and H-MMSE outperform the H-MED algorithm. On the other hand, the modified metrics employed in the H-KNN and H-MMSE algorithms cannot handle the corrupt fingerprints that follow the *False Positive* model and the localization error increases rapidly; see Figure 4.7d. In contrast, the H-MED algorithm is able to mitigate the effect of this type of fault and outperforms the other two hybrid algorithms when $\alpha > 20\%$, while the localization error remains below 6 m even when 80% of the APs provide corrupt RSS values.



(a) KNN algorithm. (b) MED algorithm. (c) H-KNN algorithm.

Figure 4.8: Mean and standard deviation of the localization error with 73 APs (*KIOS-A* setup).

4.4.4 Scalability for Varying AP Density

We now investigate the scalability of our fault tolerant localization algorithms in terms of varying AP density. The experimental results discussed previously indicate that the proposed hybrid algorithms are very resilient to faults occurring when APs either fail or are relocated. Looking at the RSS data in the *KIOS-A dataset*, there are 73 APs available; 9 APs are installed locally to fully cover the area of interest proving mostly strong RSS values and 64 other neighbouring APs are partially sensed providing mostly weak RSS values in the boundaries of the floor. Such conditions are observed in *public* localization systems, such as *Skyhook*, which rely not only on some local APs, but also publicly available and non-physically accessible APs that are further away and provide partial coverage. In this setup, the standard deviation of the localization error attained by the H-KNN algorithm is acceptable, even when $\alpha = 50\%$ of the APs suffer from *AP Failure* faults, as shown in Figure 4.8.

We have also evaluated our algorithms using the data in the *VTT dataset*, which are collected in another larger office environment covered by 31 APs as described in Section 4.1.3. Our findings, discussed in detail later in Section 4.5.4, show similar trends suggesting that our algorithms scale well to other setups.

Furthermore, we have also examined the case of considering fewer APs, e.g., only local APs and ignoring neighbouring weaker APs, to create a scenario of a *private* localization system, where almost every location sees all 9 local APs [85]. First, we follow the our methodology described to select the fault detection threshold γ for our H-KNN algorithm. Assuming that we can tolerate around 5% false detections when no faults are present, the appropriate value in this case is $\gamma = 40$ dBm.

In Figure 4.9, $\mathcal{E}$ is plotted for the MED, KNN, MMSE and H-KNN algorithms in the presence of *AP Failure* faults. In the fault-free case, the MMSE algorithm provides the best accuracy ($\mathcal{E} = 1.98$ m). For the KNN algorithm $\mathcal{E} = 2.08$ m, while for MED the mean error is 3.45 m. For the H-KNN hybrid algorithm $\mathcal{E} = 2.07$ m, which shows that the false detections in this case do not affect the localization accuracy. We can also observe that the KNN and MMSE algorithms exhibit similar fault tolerance and if the percentage of faulty APs is low ($\alpha \leq 10\%$, i.e. one AP is faulty) the localization error remains below 5 m for both methods, which may be acceptable for some location-based applications; see Figure 4.9a. However, beyond that point the localization error degrades sharply. For instance, when half of the APs are faulty, then for KNN $\mathcal{E}$ is increased by 8 m compared to the fault-free case, while the standard deviation of the error is around 5.5 m; see Figure 4.9b. Similar behaviour is also observed for the MMSE algorithm.

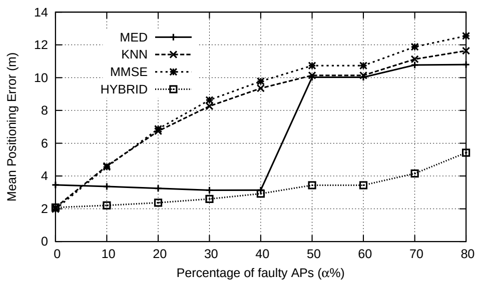
On the other hand, the MED and H-KNN algorithms exhibit similar fault tolerance in case $\alpha \leq 40\%$. The MED algorithm provides less accurate location estimates in the fault-free case, but $\mathcal{E}$ and the standard deviation of the error remain almost unaffected (around 3.30 m and 2.50 m, respectively) by the presence of *AP Failure* faults, as shown in Figure 4.9c. For the H-KNN algorithm, $\mathcal{E}$ is only increased by 0.85 m when α grows up to 40%, while the standard deviation is 2.44 m at that point, providing the same level of accuracy with MED; see Fig. 4.9d. However, for the MED algorithm, $\mathcal{E}$ explodes when $\alpha \geq 50\%$. This is due to the fact that the median-based algorithm requires that at least half of the APs provide uncorrupted RSS values. This behaviour was also reported in [69]. This is not the case for the H-KNN algorithm for which $\mathcal{E}$ remains well below 6 m, even when 80% of the APs are faulty.

To summarize, we observe that the same conclusions hold in all three different setups. Thus, our results are independent of the AP density or application specific AP layout and can be easily applied to other environments by collecting some test data, selecting the appropriate fault detection threshold, as described in Section 4.3, and then using the hybrid algorithms for localization.

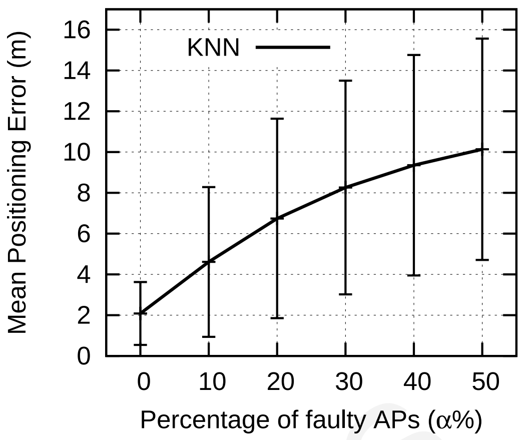
4.4.5 Discussion

We show the effectiveness of the hybrid algorithms in the case of *AP Failure* and *AP Relocation* faults with a simple numerical example.

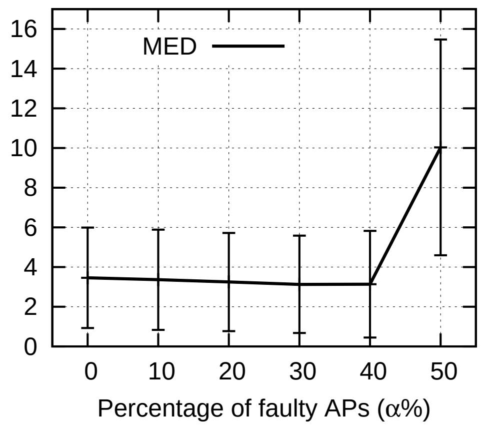
Assume that we have collected RSS measurements from 5 APs at 6 distinct locations ℓ_i, $i = 1, \ldots, 6$ (Figure 4.10a) and the fingerprints in the radiomap are shown in Figure 4.10b. Now, assume that in the fault-free case the fingerprint observed at the unknown location ℓ during localization (shown with x-mark) is $\mathbf{s} = [-69, -33, -56, -77, -31]$. By using the



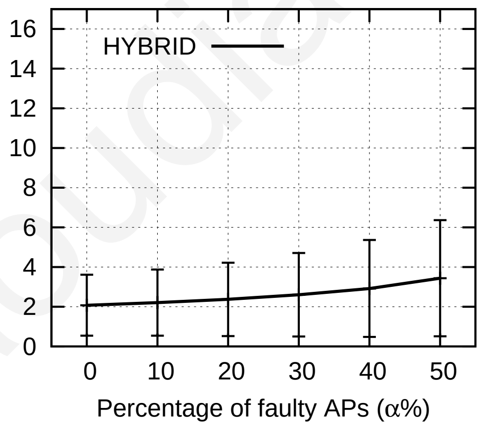
(a) Fault tolerance.



(b) KNN algorithm.



(c) MED algorithm.

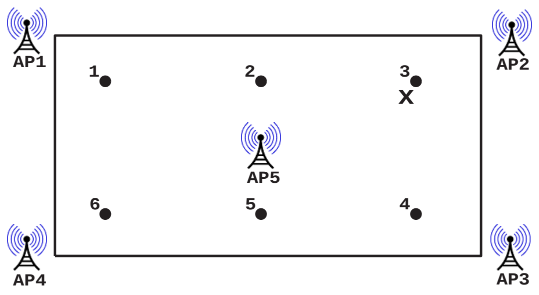


(d) H-KNN algorithm.

Figure 4.9: Mean and standard deviation of the localization error with 9 APs (*KIOS-A* setup).

Euclidean metric of equation (4.6) we obtain the ordering $\{\ell_3, \ell_2, \ell_4, \ell_5, \ell_1, \ell_6\}$ for the candidate locations with respect to increasing RSS distance. Thus, the standard NN algorithm would correctly determine ℓ_3 as the user location. Notice that by using the modified metric of equation (4.7) we still obtain exactly the same ordering. Assuming that AP$_2$ has failed and is not sensed at the unknown location, the user would observe the corrupt fingerprint $\mathbf{s}' = [-69, \mathbf{NaN}, -56, -77, -31]$. In this case using equation (4.6) to calculate the RSS distances would result in the wrong ordering $\{\ell_5, \ell_4, \ell_2, \ell_6, \ell_1, \ell_3\}$, thus introducing high error in the estimated user location. (The value −90 dBm is used to handle the missing RSS value.) In contrast, the modified distance metric of equation (4.7) ignores the missing RSS value in the distance calculations and location ℓ_3 is still ranked first. This demonstrates that H-KNN outperforms the KNN algorithm under *AP Failure* faults.

Regarding *AP Relocation* faults assume that we have moved AP$_4$ close to AP$_2$, so that the corrupt fingerprint would become $\mathbf{s}'' = [-69, -33, -56, \mathbf{-33}, -31]$. In this case, the metrics of equations (4.6) and (4.7), used in KNN and H-KNN algorithms respectively, lead to the



(a) AP layout.

Location	AP_1	AP_2	AP_3	AP_4	AP_5
1	-30	-70	-80	-58	-36
2	-49	-49	-65	-65	-25
3	-70	-30	-58	-80	-36
4	-80	-58	-30	-70	-36
5	-65	-65	-49	-49	-25
6	-58	-80	-70	-30	-36

(b) Corresponding RSS radiomap.

Figure 4.10: Example application of the hybrid fault tolerant algorithms.

wrong ordering $\{\ell_5, \ell_2, \ell_3, \ell_6, \ell_4, \ell_1\}$. On the other hand, our H-MED algorithm is not severely affected because the median-based metric of equation (4.5) can tolerate the erroneous RSS value and location ℓ_3 is ranked first.

Regarding probabilistic algorithms, the H-MMSE algorithm is more resilient to *AP Failure* faults than the standard MMSE algorithm because it uses the appropriately modified distance metric of equation (4.9), instead of (4.8). H-MMSE is not better for handling a specific type of fault and exhibits similar behaviour with H-KNN, as shown in Figure 4.7. Thus, H-MMSE should be preferred for fingerprinting systems that rely on the MMSE algorithm (or any other probabilistic approach) to exploit the likelihoods that are computed during localization.

4.5 Fault Tolerant Localization with SNAP

The SNAP algorithm described in Section 3.2.2 addresses the problem of event localization in binary sensor networks and demonstrates some desirable properties, such as low complexity and fault tolerance. In this section, we build upon the SNAP algorithm to develop a solution that is resilient to AP faults in WLAN fingerprinting.

Firstly, we adapt the SNAP algorithm to the WLAN setup using only information of whether an AP is sensed during localization or not. Secondly, we show how the accuracy of SNAP can be improved by introducing the idea of zones to exploit the discrete RSS levels. This algorithm achieves a level of accuracy that is comparable to other well-known localization methods, but is considerably simpler and much faster, which is desirable for low power mobile devices in order to save valuable energy. Finally, we investigate the fault tolerance of SNAP against other algorithms for a variety of fault or attack scenarios and present a variant of SNAP that exhibits smooth performance degradation, as the percentage of faulty APs is increased [75].

4.5.1 Localization with SNAP using Binary Data

We exploit the available RSS fingerprints by utilizing only binary information, i.e., an AP is either sensed in the online phase or not. During localization, the currently observed fingerprint $\mathbf{s}$ contains RSS values from a subset S of the available APs. In this case, the SNAP algorithm employs three main components to derive the unknown user location.

1. *Region of Coverage (RoC)*: In the offline phase, we determine the Region of Coverage $RoC_j,\ j = 1, \dots, n$ for all APs, based on the reference data in the radiomap.

2. *Likelihood Matrix $\mathcal{L}$*: In the online phase, each element in the $l \times n$ matrix $\mathcal{L}$ is updated to reflect the contribution of the j-th AP to the reference locations $\ell_i \in RoC_j$. Every AP that is sensed in the currently observed fingerprint $\mathbf{s}$ adds a positive one $+1$ contribution to the elements of $\mathcal{L}$ that correspond to the locations inside its respective RoC. On the other hand, every AP that is not sensed in $\mathbf{s}$ adds a negative one -1 contribution to the elements of $\mathcal{L}$ that correspond to the locations inside its respective RoC. Formally, the elements of $\mathcal{L}$ are obtained by

$$\mathcal{L}(i,j) = \begin{cases} +1 & j \in S \text{ AND } \ell_i \in RoC_j \\ -1 & j \notin S \text{ AND } \ell_i \in RoC_j \\ 0 & \ell_i \notin RoC_j \end{cases}. \tag{4.10}$$

   Then, for each location $\ell_i,\ i = 1, \dots, l$ we calculate the likelihood value LV_i of the user being located at ℓ_i by summing the contributions of all APs

$$LV_i = \sum_{j=1}^{n} \mathcal{L}(i,j). \tag{4.11}$$

3. *Location Estimation*: The maximum of the likelihood values points to the estimated location given by

$$\widehat{\ell}(\mathbf{s}) = \arg\max_{\ell_i \in L} LV_i. \tag{4.12}$$

   If more than one reference locations ℓ_i have the same maximum value LV_i, then the estimated location is the mean of the corresponding locations.

Example: The positive or negative contributions of a single AP (triangle) on the locations that this AP covers (dots) are shown in Figure 4.11a and Figure 4.11b, respectively. We now illustrate the application of SNAP algorithm in a simple scenario, where we consider only four APs in our reference data; see Figure 4.11c. During localization, the user resides in an

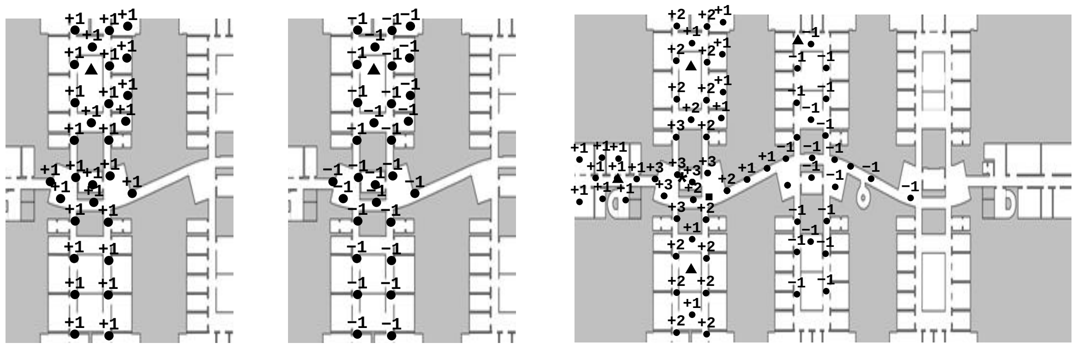

(a) AP sensed. (b) AP not sensed. (c) Example application.

Figure 4.11: Fingerprinting localization in WLAN using SNAP with binary data.

unknown location (square) and the observed fingerprint **s** contains RSS values from three APs, while the AP installed in the top middle wing is not sensed. The three APs that are sensed, add a positive one $+1$ contribution to the elements of $\mathcal{L}$ that correspond to the locations inside their respective *RoC*. The AP that is not sensed adds a negative one -1 contribution to the elements of $\mathcal{L}$ that correspond to the locations inside its *RoC*. The resulting likelihood values, after adding and subtracting the contributions of all APs, are shown in Figure 4.11c. The user location is estimated (shown with a star) as the mean of the reference locations that have the same maximum likelihood value $+3$.

4.5.2 Improving Accuracy with RSS Levels

The original SNAP algorithm uses only binary information and thus it is not expected to provide a high level of accuracy. We can further improve the performance by taking into account the information about RSS levels. The idea is that if an AP is sensed during localization, then the user is more likely to reside in the locations inside the *RoC* of a specific AP that have similar RSS values to the observed RSS value.

Based on the reference data we may determine the range of RSS values and let s_{min} and s_{max} denote the minimum and maximum RSS values, respectively. We divide this range of RSS values into N_z subranges, i.e., non overlapping equally spaced intervals, and the m-th interval Z_m is given by

$$Z_m = \Big[s_{min} + (m-1)A_r,\ s_{min} + mA_r\Big],\ m = 1,\dots,N_z, \tag{4.13}$$

where $A_r = \frac{s_{max}-s_{min}}{N_z}$.

We can now define the Zone of Coverage (ZoC) of an AP.

Definition 7. *The Zone of Coverage denoted $ZoC_{mj} \subseteq RoC_j$, $m = 1,\dots,N_z$ and $j = 1,\dots,n$,* *is the subset of reference locations ℓ_i where the j-th AP is sensed during the collection of reference RSS fingerprints and $r_{ij} \in Z_m$.*

The zones ZoC_{mj} for all available APs are determined prior to localization and essentially each RoC_j is divided into N_z zones so that $RoC_j = \bigcup_{m=1}^{N_z} ZoC_{mj}$. The modified SNAP algorithm, denoted as SNAPz, incorporates the notion of zones and the elements of $\mathcal{L}$ are now obtained by

$$\mathcal{L}(i,j) = \begin{cases} +1 & j \in S \text{ AND } \ell_i \in ZoC_{mj} \\ 0 & j \in S \text{ AND } \ell_i \in ZoC_{(m-1)j} \cup ZoC_{(m+1)j} \\ -1 & j \in S \text{ AND } \ell_i \in RoC_j - \bigcup_{k=m-1}^{m+1} ZoC_{kj} \\ -1 & j \notin S \text{ AND } \ell_i \in RoC_j \\ 0 & \ell_i \notin RoC_j \end{cases} . \quad (4.14)$$

Using this rule, every AP that is sensed in the currently observed fingerprint **s** adds a positive one $+1$ contribution only to those elements of $\mathcal{L}$ that correspond to locations inside the appropriate zone ZoC_{mj}. A zero 0 contribution is added to the locations inside the neighbouring zones, i.e., $ZoC_{(m-1)j}$ and $ZoC_{(m+1)j}$, while a negative one -1 contribution is added to the locations inside the remaining zones.

The intuition is that when an AP is sensed in the online phase with certain RSS value, then the user resides with high probability in the zone where the reference locations have similar RSS values. Due to the noise disturbing the RSS values the user may be located with some probability in the neighbouring zones. Finally, the user is located with low probability in the other zones, where the reference locations have RSS values that are very dissimilar to the observed RSS value. We define $ZoC_{0j} = ZoC_{(N_z+1)j} = \emptyset$ to handle the boundary conditions for $m = 1$ and $m = N_z$ in equation (4.14).

4.5.3 On the Fault Tolerance of SNAP

In the SNAP algorithm, an AP contributes to the location estimation whether it is sensed in the observed fingerprint **s** during localization or not. This can be very effective in the fault-free case, however, in case of faults SNAP may not be able to provide adequate performance. For instance, if faults occur during localization that follow the *AP Failure* model, then a subset of the APs that would otherwise be present in **s**, are no longer sensed. Thus, the negative contributions of these APs may introduce high errors in the estimated user location. We modify the SNAP algorithm in order to ignore the negative contributions of the failed APs and in this case the elements of $\mathcal{L}$ are obtained by

$$\mathcal{L}(i,j) = \begin{cases} +1 & j \in S \text{ AND } \ell_i \in RoC_j \\ 0 & \ell_i \notin RoC_j \end{cases} . \tag{4.15}$$

We incorporate the idea of zones into this modified algorithm to build a fault tolerant variant, denoted as SNAPft-z.

4.5.4 Performance Evaluation

We use the *VTT dataset* to evaluate the performance of SNAPz and SNAPft-z algorithms with respect to accuracy for varying number of zones in the fault-free case and resilience to AP faults, respectively. In the *VTT dataset*, we found that $s_{min} = -101$ dBm and $s_{max} = -34$ dBm. This range is divided in intervals of size $\frac{67}{N_z}$ dBm and each zone contains the locations with RSS values that fall into the respective interval.

In the case of faults, we use the fault models introduced in Section 4.2 and investigate fault tolerance with respect to the localization accuracy, as the percentage of faulty APs is increased. Moreover, we compare the SNAP-based fingerprinting algorithms to the fault tolerant algorithms H-KNN, H-MMSE and H-MED presented in Section 4.4. We apply our fault models to corrupt the original test data and the results are averaged over 20 runs using randomly selected subsets of faulty APs in each run.

Localization Accuracy of SNAPz

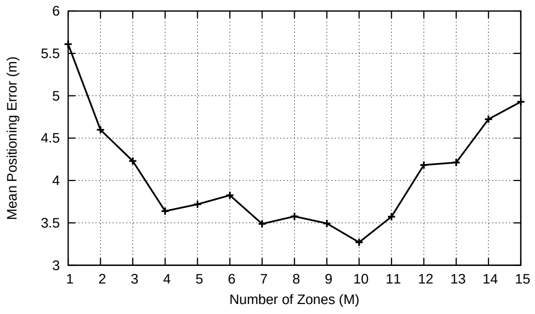
The mean positioning error ($\mathcal{E}$) pertaining to all test data is plotted in Figure 4.12a, as a function of the number of zones. If N_z is small, then each zone contains many reference locations. Thus, when a zone is "activated" during localization, then more locations are likely to be used in the location estimation and the error is increased. Note that in case $N_z = 1$, SNAPz is equivalent to the SNAP algorithm that uses only binary data. On the other hand, if N_z is large, then each zone contains only few locations and due to the noise in the RSS values the wrong zone may be activated during localization, leading to accuracy degradation.

The curve indicates that the highest level of accuracy is achieved for $N_z = 10$ zones, however $\mathcal{E}$ does not vary significantly for $4 \leq N_z \leq 11$. For reasons that are related to fault tolerance and will become clear shortly, we select $N_z = 4$ and this value is used for the rest of the experiments.

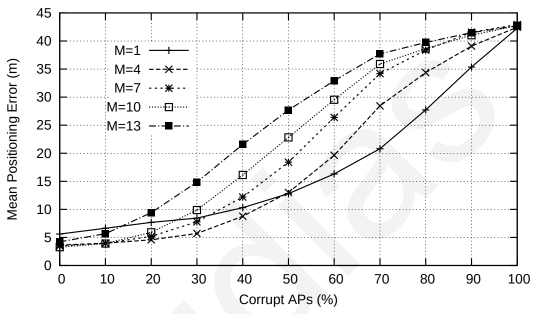
In the following, we further investigate the localization accuracy of SNAPz and the statistics for the localization error are summarized in Table 4.1. Results indicate that the MMSE al-

Table 4.1: Localization Error [m] of the SNAPz Algorithm

	Mean	Median	Std	Min	Max
KNN	2.70	2.39	1.61	0.16	8.78
MMSE	2.46	2.18	1.63	0.09	8.99
SNAPz	3.64	3.37	2.41	0.06	13.21



(a) Localization accuracy.



(b) Fault tolerance.

Figure 4.12: Performance of SNAPz for varying number of zones.

gorithm achieves the highest level of accuracy ($\mathcal{E} = 2.46$ m), followed by KNN ($\mathcal{E} = 2.70$ m). SNAPz provides less accurate location estimates and the mean error is increased by around 1 m compared to other algorithms. However, this accuracy degradation is acceptable and $\mathcal{E} = 3.64$ m is adequate for most indoor location-based services and applications.

Computational Complexity of SNAPz

Next, we investigate the estimation time of the localization algorithms by using a Matlab implementation on an Intel Pentium 4 processor 3.6 GHz with 1 GB RAM, while the execution times are averaged over 100 runs using the test data. The number of computations and the time required by each method are summarized in Table 4.2.

SNAPz does not require heavy computations and one location estimate takes 0.49 ms, which is 1.6 and 3.5 times lower compared to KNN and MMSE algorithms, respectively. Therefore, the SNAPz method can extend the battery life of mobile devices, especially when frequent localization requests or tracking applications are considered.

Fault Tolerance of SNAPz and SNAPft-z

The performance of SNAPz under the *AP Failure* model for varying number of zones is plotted in Figure 4.12b. In the fault-free case and when less than 50% of APs are corrupted, using $N_z = 4$ zones provides a high level of accuracy and the performance degrades smoothly. In

Table 4.2: Computational Complexity of Localization Methods

	additions	multiplications	exp	sorts	time (ms)
KNN	$(2n-1)l$	nl	0	l	1.25
MMSE	$(n+3)l-3$	$(2n+4)l$	nl	0	2.18
SNAPz	$(n-1)l$	0	0	l	0.49

l: # of reference locations, n: # of APs, sorts: # of floats to be sorted

case more than half of the APs are corrupted, using $N_z = 1$ zone improves the fault tolerance of SNAPz. Using more zones has a negative effect and $\mathcal{E}$ increases rapidly. Similar behaviour was observed for the other fault models. Thus, we use $N_z = 4$ zones in SNAPz algorithm that is a good trade-off between accuracy and fault tolerance. Interestingly, we found that $N_z = 4$ zones is a good option for SNAPft-z as well. In the following, we do not consider the SNAPz algorithm because our findings indicate that it is less fault tolerant than SNAPft-z [75].

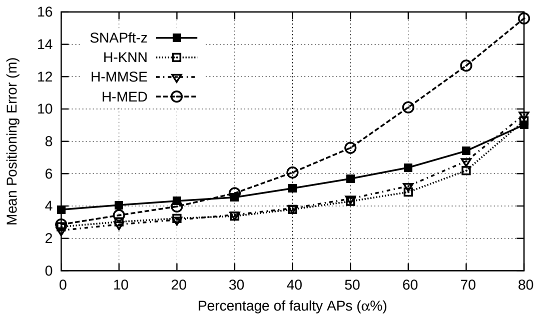
We apply all fault models described in Section 4.2 to simulate different types of faults and compare the SNAPft-z algorithm against the H-KNN, H-MMSE and H-MED algorithms with respect to fault tolerance. The H-KNN, H-MMSE and H-MED algorithms can be easily applied to the experimental setup of the *VTT dataset*. In particular, we use the methodologies described in Section 4.3 to obtain the appropriate fault detection thresholds for this setup, i.e., $\gamma = 61$ dBm for H-KNN and H-MED and $\gamma = 42$ for H-MMSE.

Under the *AP Failure* model, the H-MED does not provide adequate fault tolerance, especially when the percentage of faulty APs exceeds 30%, as shown in Figure 4.13a. Beyond that point, the SNAPft-z algorithm proves to be more resilient to faults, however its performance is inferior compared to the H-KNN and H-MMSE algorithms.

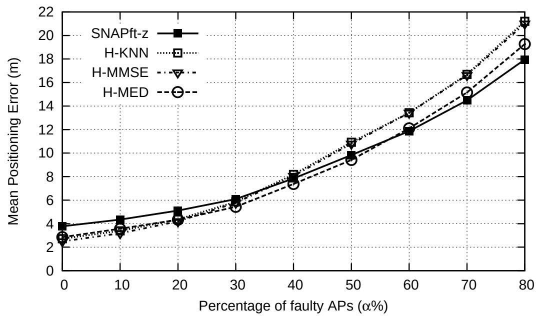
In the case of *AP Relocation* faults, SNAPft-z has higher fault tolerance than H-KNN and H-MMSE when $\alpha \geq 30\%$ and its performance is similar to the H-MED algorithm; see Figure 4.13b.

When the *False Negative* model is assumed, we observe similar behaviour with *AP Failure* faults. SNAPft-z exhibits high fault tolerance and outperforms the H-MED algorithm when more that 40% of the APs are corrupted, as shown in Figure 4.13c. Still the H-KNN and H-MMSE algorithms can handle better this type of fault.

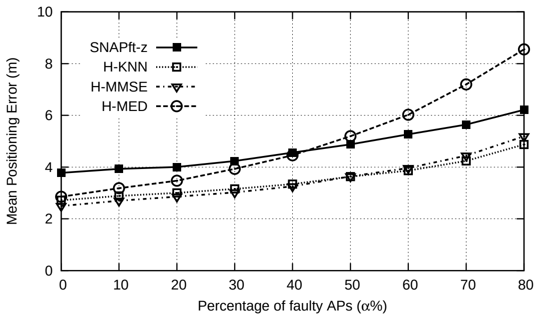
Faults under the *False Positive* model cause severe performance degradation when the H-KNN and H-MMSE algorithms are considered, as shown in Figure 4.13d. This confirms our previous findings (Figure 4.7d) using the *KIOS-A dataset*. In this case, the SNAPft-z algorithm delivers high fault tolerance and outperforms the H-MED algorithm when more that 60% of



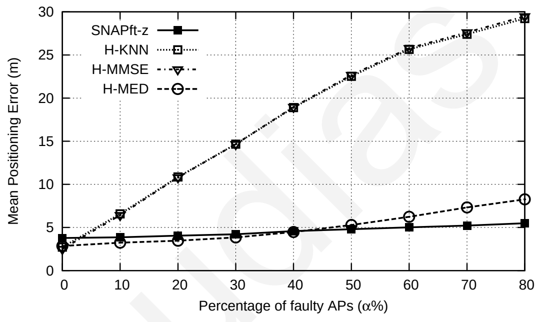
(a) *AP Failure* model.



(b) *AP Relocation* model.



(c) *False Negative* model.



(d) *False Positive* model.

Figure 4.13: Fault tolerance of SNAPft-z under various fault models.

the APs provide corrupt values.

Overall, the SNAPft-z algorithm is less accurate compared to H-KNN, H-MMSE and H-MED in the fault-free case, as well as when the percentage of faulty APs remains low. This is expected because the SNAPft-z algorithm does not exploit the full range of RSS values in the computations, contrary to the other algorithms. Under *AP Failure* and *False Negative* faults, SNAPft-z attains adequate fault tolerance, however it is outperformed by H-KNN and H-MMSE. On the other hand, SNAPft-z is a very good option when *AP Relocation* and *False Positive* faults are present in the fingerprinting system. To summarize, SNAPft-z should be preferred for mitigating any type of fault on mobile devices with limited processing and energy resources, due to its low computational complexity. Moreover, it is the best candidate solution for fault tolerant localization in the presence of *False Positive* faults.

4.6 Chapter Summary

The presence of faults during localization, e.g., caused by AP failures or relocations and malicious attacks, can lead to significant accuracy degradation. In this chapter we focus on fault detection and mitigation in order to improve the fault tolerance of localization algorithms.

To this end, we start by introducing realistic fault models, namely the *AP Failure*, *AP relocation*, *False Negative* and *False Positive* models, that capture different types of abnormal behaviour with respect to AP functionality in fingerprinting localization systems owing to hardware failures or malicious attacks. We discuss the feasibility of the underlying attack or the occurrence probability of the relevant failure and describe how these new models can be simulated using the original test data.

Subsequently, we develop robust fault detection schemes using RSS distance-based or likelihood-based indicators. By coupling these detection mechanisms with properly modified metrics in the underlying localization algorithm we derive hybrid algorithms that significantly improve the resilience of the standard fingerprinting algorithms to faults.

In particular, we propose three new fault tolerant algorithms, namely H-KNN, H-MMSE and H-MED. The H-KNN algorithm combines the RSS distance-based fault detection mechanism with a modified distance metric and greatly improves the fault tolerance of the original KNN algorithm, when APs fail accidentally or maliciously. The H-MMSE algorithm combines our likelihood-based fault detection mechanism with a modified likelihood metric to increase the resilience of the MMSE algorithm to *AP Failure* faults. The H-KNN and H-MMSE algorithms exhibit high fault tolerance in the presence of *AP Failure* and *False Negative* faults, however they are less effective under other types of faults.

To this end, the H-MED hybrid algorithm switches between the Euclidean and the median-based distance metrics to alleviate the effect of *AP Relocation* faults, while at the same time improves the accuracy of the median-based localization algorithm in the fault-free case.

On a different line, we build upon the SNAP algorithm, which exhibits high resilience to sensor faults in binary WSNs, and adapt it to WLAN setups. We introduce an implementation of the SNAP algorithm for RSS fingerprinting in WLAN that is simple and time efficient, i.e., by using only binary information of whether an AP is sensed or not. This algorithm is modified to derive the SNAPft-z algorithm that reduces the localization error and improves the resilience to AP faults. Experimental results indicate that SNAPft-z achieves high fault tolerance, especially when *False Positive* faults are present, while maintaining low computational complexity.

Chapter 5

Localization for Diverse Devices

5.1 Background

5.1.1 Problem Formulation & Definitions

For the purposes of this chapter we extend the formulation of the RSS fingerprinting problem, introduced in Section 4.1.1. In particular, these extensions allow to deal with the device diversity problem in a formal way for the following two cases:

1. The device carried during the *online* phase may differ from the device that was used to collect the reference data for building the radiomap

2. The reference data in the radiomap may be collected with multiple devices in a crowdsourcing fashion

In the *offline* phase, we consider a set of predefined reference locations $\{L : \ell_i = (x_i, y_i),\ i = 1, \ldots, l\}$ on a grid over the localization area. We collect RSS measurements from n APs with a set of heterogeneous devices $D^{(m)},\ m = 1, \ldots, M$. Device m visits a subset of the reference locations $\{L^{(m)} : \ell_i = (x_i, y_i),\ i = 1, \ldots, l^{(m)}\}$, so that $L^{(m)} \subseteq L$ and $L = \bigcup_{m=1}^{M} L^{(m)}$. A reference fingerprint $\mathbf{r}_i^{(m)} = [r_{i1}^{(m)}, \ldots, r_{in}^{(m)}]^T$ associated with location ℓ_i is a vector of RSS samples and $r_{ij}^{(m)}$ denotes the RSS value from the j-th AP collected using device $D^{(m)}$. These fingerprints are contained in the device-specific radiomap $\mathbf{R}^{(m)} \in \mathbb{Z}^{-}_{n \times l^{(m)}}$ that may partially cover the area, while all devices contribute their respective radiomaps for building the crowdsourced radiomap $\mathbf{R} \in \mathbb{Z}^{-}_{n \times l}$ that covers the whole area. This is done by aggregating the RSS values for each AP across all contributing devices M_i at location ℓ_i, where $1 \le M_i \le M$, according to

$$r_{ij} = \frac{1}{M_i} \sum_{m=1}^{M_i} r_{ij}^{(m)}. \tag{5.1}$$

Note that this formulation includes the extreme cases where the device-specific radiomaps correspond to non-overlapping contributions ($M_i = 1, \forall i$) in the localization area, as opposed to fully overlapping contributions ($M_i = M, \forall i$).

In the *online* phase, we exploit the crowdsourced radiomap $\mathbf{R}$ to obtain a location estimate $\widehat{\ell}$, given a new fingerprint $\mathbf{s} = [s_1, \ldots, s_n]^T$ measured at the unknown location ℓ by the user-carried device $D^{(m')}, m' \in \{1, \ldots, M'\}$ with $M' > M$.

5.1.2 Experimental Setup & Dataset

Our dataset contains reference data, collected at predefined reference locations with one or more WLAN-enabled mobile devices, that are used to build the required radiomap for the fingerprinting algorithms. Moreover, the datasets contain additional test data, collected at several test locations with one or more mobile devices, that are used for the validation of the developed algorithms.

KIOS-B Dataset

The *KIOS-B* dataset was collected at the KIOS Research Center, later than the *KIOS-A* dataset described in Section 4.1.3, using exactly the same reference and test locations. This new dataset differs from the *KIOS-A* dataset only with respect to the number of devices and the number of collected samples. In particular, apart from the HP iPAQ hw6915 PDA device, we have also used 4 other different mobile devices for our data collection, namely an Asus eeePC T101MT laptop running Windows 7, an HTC Flyer Android tablet and two other Android smartphones (HTC Desire and Samsung Nexus S)[1]. The data collection for the Android devices was conducted with our award winning *Airplace* logging and positioning platform[2] [78–80, 82]. Specifically, we used the *Airplace Logger* application to collect RSS values from the surrounding APs at several reference and test locations, as follows. The floorplan map is displayed on the Android device within the *Airplace Logger* user interface, enabling the users to select their current location by clicking on the map and then click the on-screen buttons to initiate and end the logging process.

For our reference data we recorded fingerprints, which contain RSS measurements from the surrounding APs, at 105 distinct reference locations by carrying all 5 devices at the same time. The RSS measurements come from 9 APs installed locally to provide full coverage. More-

[1]The *KIOS-B* dataset is available to download at `http://goo.gl/u7IoG`

[2]*Airplace* has been released as an open source project and is available at `http://goo.gl/3uaGKe`

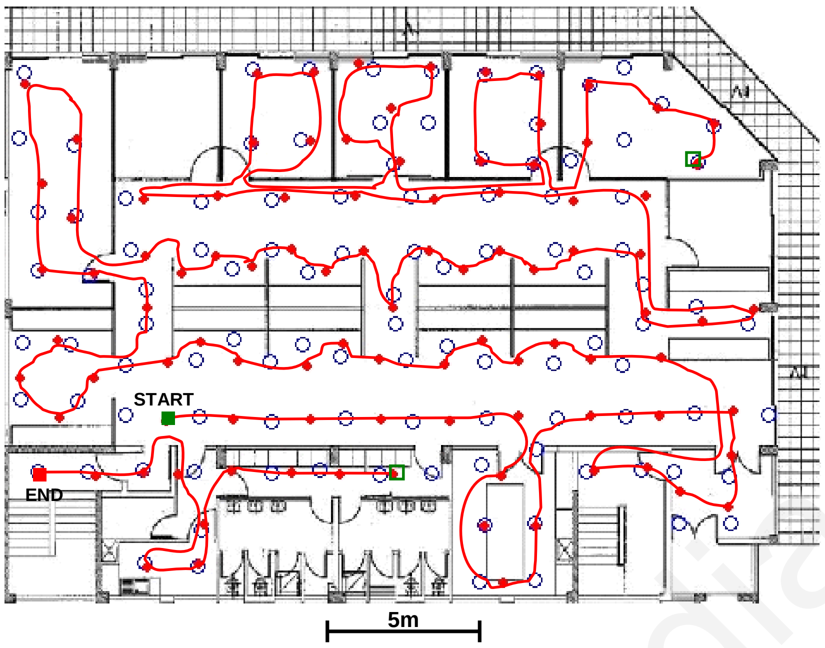


Figure 5.1: Reference and test locations in the *KIOS-B* dataset.

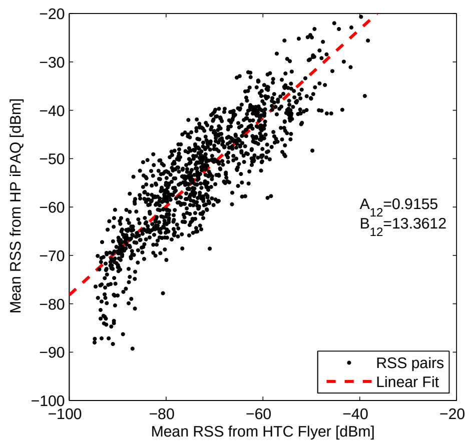
over, there is a varying number of neighbouring APs that can be sensed in different parts of the floor and in some locations more than 60 APs could be sampled, depending on the device. A total of 2100 training fingerprints, corresponding to 20 fingerprints per reference location, were collected with each device. These data are used to build device-specific radiomaps by calculating the mean value RSS fingerprint that corresponds to each reference location.

Two weeks later, we collected additional test data at 96 locations by carrying all 5 devices simultaneously, while 10 fingerprints were recorded at each test location. The test locations (red dots), most of which do not coincide with the reference locations (blue circles), lie along a path marked with red line that has two segments (green square indicates the end of the first segment and the beginning of the second segment), as shown in Figure 5.1.

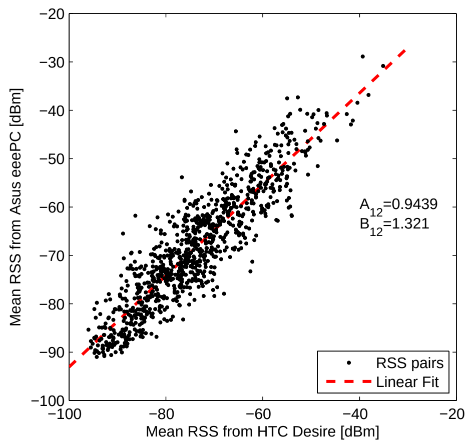
5.2 Revisiting Manual Device Calibration

Several experimental studies have reported a linear relation between the RSS values reported by heterogeneous devices [52, 62, 116]. Thus, if a sufficient number of colocated RSS pairs (i.e., collected at the same location and time with two different devices) is available, then the linear parameters can be estimated through standard least squares fitting. To put it formally, for two devices $D^{(1)}$ and $D^{(2)}$ we use the RSS data in the respective radiomaps to compute the parameters by

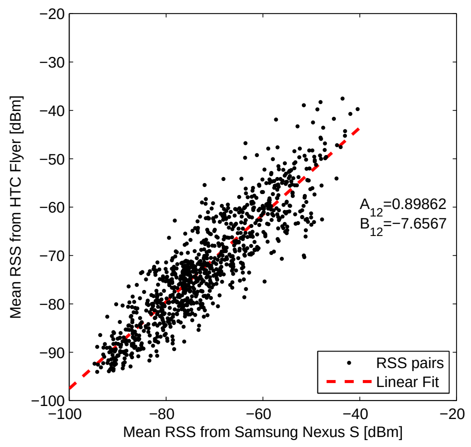
$$r_{ij}^{(2)} = A_{12} r_{ij}^{(1)} + B_{12}, \tag{5.2}$$



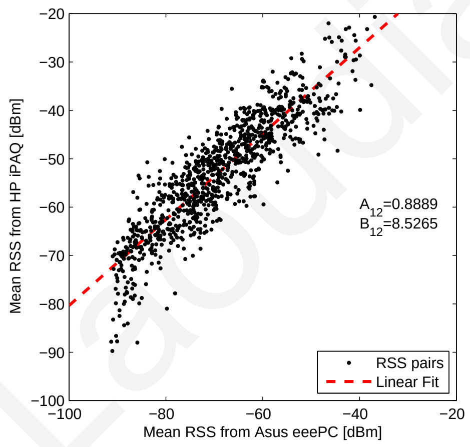
(a) HP iPAQ – HTC Flyer pair.



(b) Asus eeePC – HTC Desire pair.



(c) HTC Flyer – Samsung Nexus S pair.



(d) HP iPAQ – Asus eeePC pair.

Figure 5.2: Linear relation between RSS values from diverse devices.

where $r_{ij}^{(1)}$ and $r_{ij}^{(2)}$ denote the mean RSS value at location ℓ_i from the j-th AP for $D^{(1)}$ and $D^{(2)}$ respectively, while A_{12} and B_{12} are the linear parameters for mapping the RSS values from $D^{(1)}$ to $D^{(2)}$. Some indicative correlation plots for various pairs of devices using data from our *KIOS-B* dataset are shown in Figure 5.2. These plots confirm the linear relation between the RSS values reported by heterogeneous devices and justify the effectiveness of first order polynomials for device calibration.

Assuming that the radiomap contains data from a single reference device $D^{(1)}$, an important question is how much data should be collected with a new device $D^{(m)}, m \neq 1$ to achieve a good mapping to the reference device $D^{(1)}$ and guarantee acceptable accuracy when this new device $D^{(m)}$ is used for localization. For instance, if a considerable volume of RSS data that spans the whole area of interest needs to be collected with another device $D^{(m)}$, then this new dataset may as well be used as a second radiomap that will be employed whenever device $D^{(m)}$ is carried by a user. Thus, *manual calibration* is only justified if a small number of RSS fingerprints collected at a few calibration locations suffice to obtain an adequate mapping.

Authors in [52] report that in their setup, where around 15 APs are sensed at each location, *manual calibration* is very effective when RSS fingerprints collected at three to five locations are used for fitting. Our experimental results are in agreement and in the following we provide a justification to support this observation.

Suppose that the colocated RSS pairs from two different devices $y = [r_{ij}^{(1)}, r_{ij}^{(2)}]^T$ follow a normal distribution, i.e. $y[k]|x \overset{\text{iid}}{\sim} \mathcal{N}(x, R)$, where x is a bivariate random vector and R is the covariance matrix. Roughly speaking, x and R represent the center and the shape of the cloud of RSS pairs in this 2-D space, respectively; see Figure 5.3a. In the linear fitting given by equation (5.2), the parameter B_{12} is directly related to the cloud center x, while the parameter A_{12} is related to the principal axis of the cloud shape R. For brevity, we assume that R is known, i.e. A_{12} value is fixed, and in the following we study the behaviour of the cloud center x for increasing number of RSS pairs that provides insight into the convergence of the B_{12} parameter[3].

Assuming prior distribution $x \sim \mathcal{N}(m[0], P[0])$, the posterior distribution given a series of colocated RSS pairs is $x|y[1{:}k] \sim \mathcal{N}(m[k], P[k])$ with

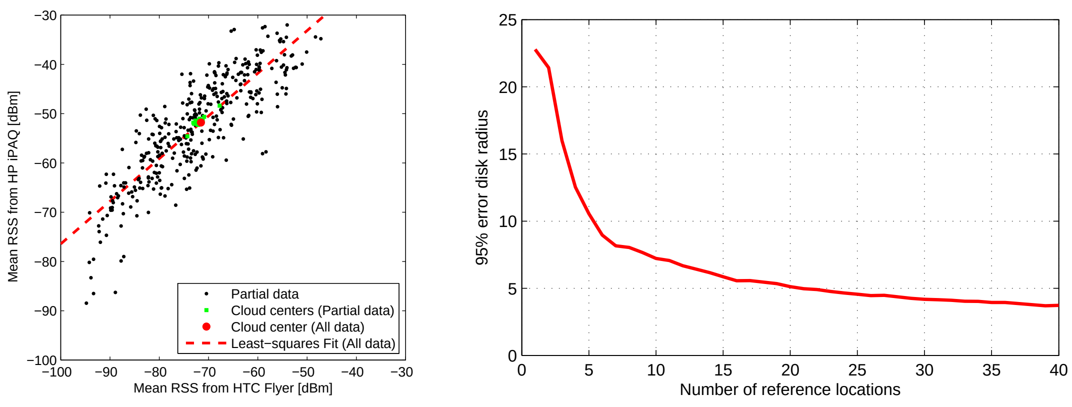
$$P[k] = P[k-1] - P[k-1](P[k-1] + R)^{-1}P[k-1] \tag{5.3}$$

$$m[k] = m[k-1] + P[k-1](P[k-1] + R)^{-1}(y[k] - m[k-1]). \tag{5.4}$$

These equations are used for updating $m[k]$ and $P[k]$ recursively [119] and improve the estimate of x by sequentially processing the RSS observations from pairs of devices. By Chebyshev's inequality, the disk with center $m[k]$ and radius $r = \sqrt{\text{trace}(P[k])/0.05}$ contains x with probability at least 95% [119]. It can be shown that $\text{trace}(P[k]) \leq \text{trace}(R)/k$ and consequently the 95% disk's radius is proportional to $1/\sqrt{k}$. As a rule of thumb, to get for example 10 times better accuracy, 100 times more data are required. We may use the sample covariance estimator to calculate R from a series of colocated RSS pairs $y[1{:}k]$ and study the 95% error disk's radius by plotting $\sqrt{\text{trace}(R)/(0.05k)}$. The center of the RSS cloud (shown in green) for increasing number of locations that contribute their mean RSS pairs to the fitting is illustrated in Figure 5.3a. Note that these locations are uniformly distributed and each location contributes 9 RSS pairs[4]. It is observed that the center x, when data from few locations are used, converges

[3]The simultaneous estimation of x and R is also possible by treating them both as unknown and assuming a flat prior for x and a Wishart prior for R and then using recursive formulas for the posterior means of x and R given $y[1{:}k]$, as discussed in [25].

[4]For simplicity, in this analysis we consider only the RSS values from the 9 APs installed inside the experimentation area.



(a) Centres of the RSS cloud. (b) Convergence of the 95% error disk radius.

Figure 5.3: Manual device calibration using a small amount of RSS data.

quickly to the center obtained when data from all 105 locations (i.e., around 945 RSS pairs) are considered. Moreover, the 95% error disk's radius decreases as the number of locations grows from 1 to 40, indicating that colocated RSS pairs from 15 to 20 locations seem to suffice; see Figure 5.3b.

The above analysis justifies the effectiveness of the *manual calibration* using a small amount of data and in practice we observed that around 5 known locations uniformly distributed inside the area of interest can provide good device calibration. The statistics of the positioning error using data from a variable number of locations (pertaining to the whole test set) are listed in Table 5.1 using the HP iPAQ and the HTC Flyer as reference and new device, respectively. These results indicate that the localization accuracy can be significantly improved when *manual calibration* is applied. For instance, the mean error decreases to 2.7 m when we use colocated RSS pairs collected with the HTC Flyer at all 105 reference locations, compared to 7.6 m when no device calibration is applied. Interestingly, we observe that using 20 or only 5 locations (i.e., 180 or 45 colocated RSS pairs) for *manual calibration* has marginal effect on the localization error. Note that in the cases where few reference locations are considered for *manual calibration*, the results pertain to 10 experiments assuming different subsets of randomly selected locations.

These results confirm that *manual calibration* with colocated RSS pairs collected at *known* locations is a very effective approach. More importantly, when the area of interest is covered by several APs then only a few locations need to be visited with an uncalibrated device, thus reducing the time and labour overhead for calibrating a new device.

However, this approach has limited applicability in real-life applications where users enter an indoor environment, such as shopping malls, airports, etc., carrying an uncalibrated device,

Table 5.1: Positioning error [m] of the manual device calibration.

	Uncalibrated	All	20	5
Mean	7.6	2.7	2.7±0.0	2.9±0.2
Median	7.3	2.3	2.3±0.0	2.4±0.1
67% CDF	9.4	3.0	3.0±0.0	3.1±0.3
95% CDF	15.7	6.2	6.3±0.3	7.0±0.8
Max	19.1	16.2	15.0±1.3	14.6±1.1

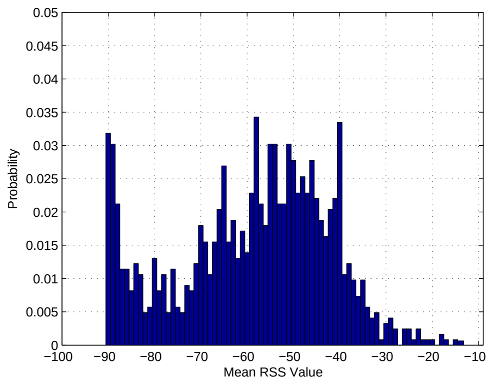
because they have to be guided to specific *known* locations for collecting RSS data. This implies that a user is already familiar with the area of interest, which is usually not the case. Moreover, a considerable data collection effort is still required by the user prior to localization. Our approach described in the following is to exploit histograms of RSS values collected with the reference and user-carried device in order to develop a fully automatic approach that does not require any user intervention.

5.3 Histogram-based Device Calibration

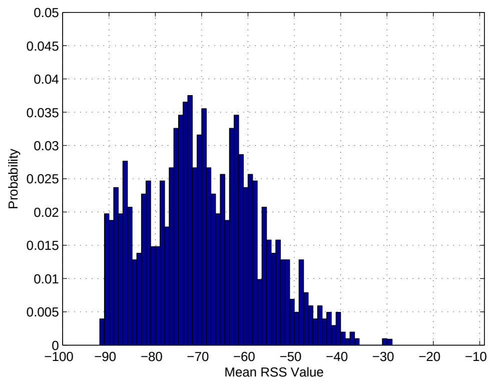
5.3.1 RSS Histograms

The RSS histograms of three different devices are shown in Figure 5.4. These histograms correspond to the mean RSS values from all available APs collected at all 105 reference locations. The first observation is that two histograms may differ significantly with respect to the range of RSS values, as well as the probability of each RSS value, as in the case of HP iPAQ and HTC Flyer. On the other hand, the respective histograms for some device pairs can be quite similar, as in the case of Asus eeePC and HTC Flyer. Note that compared to our results reported in [87], where we considered only the RSS values from the 9 APs installed locally, the histograms in Figure 5.4 are left-skewed because there is a large number of weak RSS values recorded from APs that are located far from the user. This is most evident in the histogram of the iPAQ device, which features a more sensitive WLAN adapter and is able to sense very distant APs (Figure 5.4a).

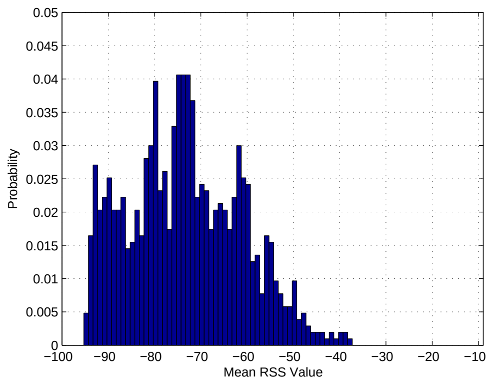
Assuming that the relation between the RSS values reported by diverse devices is linear, there is a simple way to exploit the histograms that are accumulated on each device for some time. Specifically, one can deduce the extreme values from the respective RSS histograms and



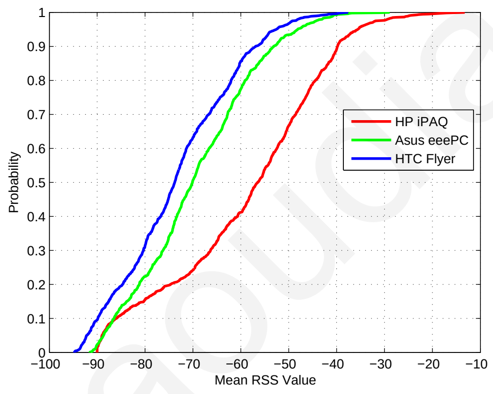
(a) HP iPAQ.



(b) Asus eeePC.



(c) HTC Flyer.



(d) Empirical CDFs.

Figure 5.4: Histograms of the mean RSS values and corresponding empirical CDFs.

then obtain the linear fitting parameters from the following system of equations

$$\mathbf{S}^{(2)} = A_{12}\mathbf{S}^{(1)} + B_{12}, \tag{5.5}$$

where $\mathbf{S}^{(1)} = [s_{min}^{(1)}\ s_{max}^{(1)}]^T$ and $\mathbf{S}^{(2)} = [s_{min}^{(2)}\ s_{max}^{(2)}]^T$ denote the vectors that contain the minimum and maximum RSS values in the histograms of devices $D^{(1)}$ and $D^{(2)}$, respectively. This method is very appealing due to its low complexity, however the calibration performance can deteriorate significantly under some conditions, as shown by the experimental results in Section 5.3.3.

5.3.2 Self-calibration Method

The relation among the RSS histograms of different devices is also reflected in the equivalent empirical CDF (eCDF), as shown in Figure 5.4d. We have observed that the eCDF of the raw RSS values, recorded while walking around with a particular device for a few seconds, resembles the respective eCDF of the mean RSS values collected with the same device at several uniformly distributed *known* locations. This implies that we may exploit these eCDFs to perform device calibration during localization. The main idea in our self-calibration method is

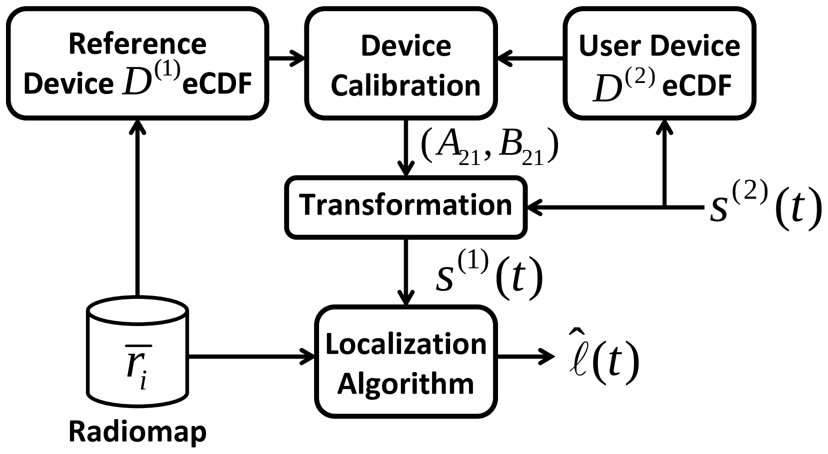


Figure 5.5: Block diagram of the device self-calibration method.

the use of RSS histograms to obtain a mapping between the reference and various user devices, thus avoiding the bother of the *manual calibration* approach.

The block diagram of our method is shown in Figure 5.5. First, we use the existing radiomap to obtain the RSS histogram of the reference device $D^{(1)}$. Subsequently, when the user enters a building and starts localizing, the RSS values in the currently observed fingerprint $\mathbf{s}^{(2)}(t)$ are recorded simultaneously in the background in order to create and update the histogram of raw RSS values for the user-carried device $D^{(2)}$. Then, we use the RSS values that correspond to specific percentiles of the eCDF to fit a linear mapping of the form in equation (5.2) between the user and reference devices. Subsequently, the parameters (A_{21}, B_{21}) are used to transform the RSS values observed with the user device and obtain the corresponding fingerprint $\mathbf{s}^{(1)}(t)$, where $s_j^{(1)}(t) = A_{21}s_j^{(2)}(t) + B_{21},\ j = 1, \ldots, n$. The resulting fingerprint $\mathbf{s}^{(1)}(t)$ is compatible with the radiomap and finally the unknown location $\hat{\ell}(t)$ can be estimated with any fingerprinting algorithm. The *Device Calibration* component in our method that computes the parameters (A_{21}, B_{21}) is detailed in the following.

Let $F_1(x)$ and $F_2(x)$ denote the eCDFs of the reference and user devices, respectively. In general the CDF $F(x)$ gives the probability of observing an RSS value that is less than x, while the inverse CDF $F^{-1}(y)$ returns the RSS value that corresponds to the y-th CDF percentile. We use the RSS values that correspond to the 10-th, 20-th, …, 90-th percentiles of the eCDF to fit a least squares linear mapping between the user and reference devices and estimate the parameters (A_{21}, B_{21}) according to

$$F_1^{-1}(y) = A_{21}F_2^{-1}(y) + B_{21},\ y \in \{0.1, 0.2, \ldots, 0.9\}. \tag{5.6}$$

A formal proof on the validity of the least squares mapping in equation (5.6) that uses the inverse CDF percentile values to reveal the underlying functional relationship between the

RSS values collected with different devices can be found in Appendix F.

A question that arises is how much time is needed in practice until the user device is self-calibrated. While the user is walking, the current fingerprint $\mathbf{s}^{(2)}(t)$ contributes only a few RSS values and $F_2(x)$ does not change significantly between two consecutive samples. Thus, it is not necessary to update $F_2(x)$ every time a new fingerprint $\mathbf{s}^{(2)}(t)$ is available, but rather one can buffer the RSS values contained in a number of successive fingerprints and then update $F_2(x)$ before performing the linear fitting. We have experimentally found that the buffer size $b_s = 10$ works well in our setup, i.e., the parameters (A_{21}, B_{21}) are recalculated every 10 seconds. At the beginning, we initialize the parameters to $(A_{21}, B_{21}) = (1, 0)$, i.e., no transformation is performed, to handle the localization requests until the buffer is full and the parameters are estimated for the first time. Using a lower value for b_s does not seem to improve the localization accuracy significantly, while it introduces unnecessary computational overhead. On the other hand, increasing b_s means that the parameters are not updated frequently enough and the performance is degraded, especially at the beginning until (A_{21}, B_{21}) are estimated for the first time.

Calculating the least squares fitting parameters (A_{21}, B_{21}) is the most demanding task in our self-calibration method in terms of computational power. We propose a modification to address this issue in case such computation is costly for low-resource mobile devices. In particular, we fix $A_{21} = 1$, so that we actually fit a unit slope linear mapping and only estimate the parameter B_{21} using the following median estimator

$$B = \mathrm{med}(F_1^{-1}(y) - F_2^{-1}(y)),\ y \in \{0.1, 0.2, \ldots, 0.9\}. \tag{5.7}$$

This approach is valid because several experimental studies [64,87,138] have reported that the A values among different device pairs are usually around 1. With this modification only one, instead of two parameters, needs to be estimated and the computational overhead of the least-squares fitting in equation (5.6) is significantly reduced. More importantly, this benefit comes without compromising the performance of the self-calibration method, as the experimental results in Section 5.3.3 indicate.

5.3.3 Experimental Results

We assess the performance of the proposed self-calibration method using the experimental data from all 5 devices in the *KIOS-B* dataset, described in Section 5.1.2. For simplicity we drop the subscripts in the linear fitting parameters and in the following (A, B) denotes the fitting parameters between the reference and user devices that can be any of the 5 devices considered in our dataset.

As we have visited the test locations in that dataset one after the other, for the purposes of this section, the test data can also be viewed as sampling the same path 10 times with all devices, while 1 fingerprint was recorded at each test location. Moreover, the results reported here take into account all available APs. Preliminary results based only the APs installed locally inside the experimentation area are reported in [87].

We consider two variants, namely the self-calibration method, denoted SC, that calculates the fitting parameters with equation (5.6) and the modified self-calibration method that assumes $A = 1$ and estimates B with equation (5.7), denoted SCmed. We also evaluate the histogram-based method of equation (5.5) that computes the fitting parameters using only the minimum and maximum RSS values for each device pair, referred to as MM.

In our comparison we also include some state-of-the-art calibration-free methods, such as the SSD method [101], the DIFF method [40], the HLF method [64] and the RBF method [99]. For the SSD method we select the anchor AP as the one with the least average deviation of RSS values over the whole localization area [56] and we apply the same approach to the HLF method. Finally, for completeness we report the localization accuracy of the *manual calibration* method [52, 62], denoted MC, that uses the mean RSS values collected with the user device at all 105 locations visited with the reference device, as well as the two extreme cases of No Calibration (NC) and using a Device Specific radiomap (DS) collected with each device that provide the upper and lower bound on the performance, respectively.

First, we demonstrate the efficiency of the SC method on a single route using the iPAQ radiomap, while the user carries the Flyer device. The performance of our method is illustrated in Figure 5.6a, where we have used a buffer size $b_s = 10$. We observe that in the first 10 seconds the accuracy is not adequate, because the device is still uncalibrated. While the user is walking the raw RSS values are collected in order to build the RSS histogram that will be used for the self-calibration. It is obvious that beyond that point, the user device has been automatically calibrated and the localization system delivers accuracy that is considerably better compared to the no calibration case and is much closer to the case of using a radiomap that is created from data collected with the Flyer device.

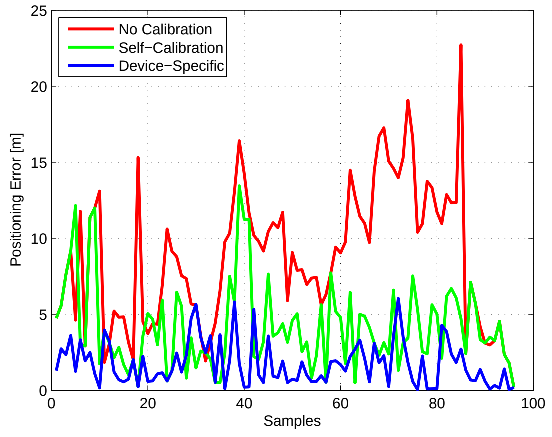
Next, we investigate the performance of the SC method in terms of the localization error attained while the user is walking. In particular, we calculate the mean positioning error $\bar{\epsilon}$ for a single route, which is defined as the distance between the estimated and actual user locations averaged over the 96 locations that comprise the testing route. By sampling the testing route 10 times, we calculate the statistics for $\bar{\epsilon}$. These statistics, pertaining to all 10 routes, are depicted as boxplots in Figure 5.6 for some indicative device pairs. On each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, while the whiskers extend to the most extreme values of $\bar{\epsilon}$, except outlier values which are shown individually as red crosses.

The first observation is that localizing without device calibration should be avoided because it may lead to significant accuracy degradation. For instance, for the iPAQ – Desire pair (Figure 5.6b) if the Desire device is not calibrated the median of the mean error $\bar{\epsilon}$ is around 10 m compared to 2.2 m in case we use a radiomap collected with the Desire device (i.e., DS method), instead of the iPAQ. Both variants of our self-calibration method (i.e., SC and SCmed) are very effective and achieve performance that is very close to the *manual calibration* approach as shown in Figure 5.6b, but with considerably less effort.

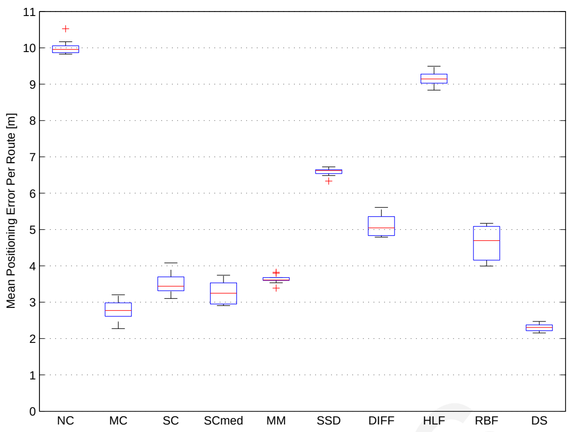
Note that for the calculation of the error statistics we have included all the initial location estimates, obtained before the parameters (A, B) have been estimated. Regarding the estimation of the parameters (A, B), the process was restarted at the beginning for each one of the 10 testing routes, i.e., for each route the parameters were initialized to $(1, 0)$ and then estimated using the proposed methodology, to guarantee the same conditions for the 10 experiments. Of course, in real life applications, these parameters can be estimated once when the user walks inside a given indoor environment for the first time and then be stored on the device for future use in the same environment.

Surprisingly, the SSD and HLF methods perform poorly for the iPAQ – Desire pair. It turns out that their performance can be improved, if we consider only the local APs, instead of all the APs in the vicinity, during localization as reported in [87]. This is a strong indication that for small-scale, fully-controlled setups, where the APs provide full coverage, the SSD and HLF methods are adequate. However, in large-scale setups, where the APs provide intermittent coverage, their performance might deteriorate. For other device pairs, the SSD and HLF methods seem to be more robust to device diversity, however they are still both outperformed by the proposed self-calibration method; see for example Figure 5.6c and Figure 5.6d. This is not the case for the other differential fingerprinting method (DIFF) that attains almost the same localization accuracy with our self-calibration methods. This will be further analysed and discussed later in Section 5.4.

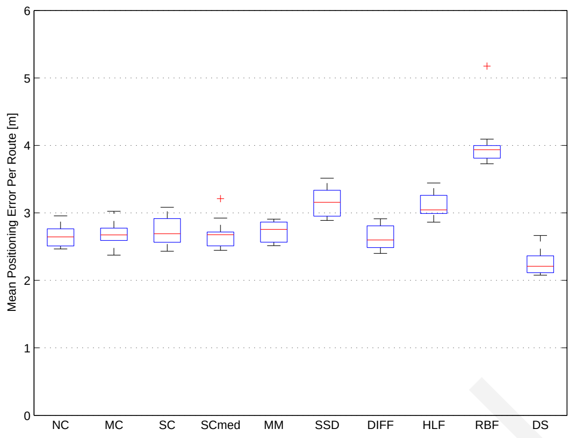
The performance of the RBF method is also very poor in most scenarios, highlighting that the fine-grain information of the RSS levels is lost when ranks are used. On the other hand, the simple histogram-based MM method seems to work well in practice for some de-



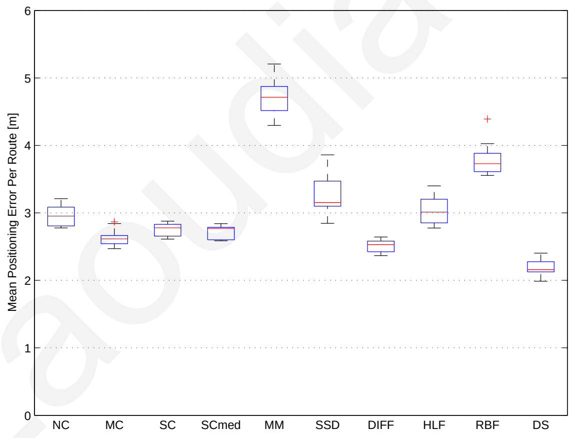
(a) Positioning error in a single route.



(b) HP iPAQ radiomap – HTC Desire user.



(c) HTC Flyer radiomap – Samsung Nexus S user.



(d) HTC Flyer radiomap – Asus eeePC user.

Figure 5.6: Localization accuracy of the self-calibration method.

vice pairs and achieves the same level of accuracy as our self-calibration method, as shown in Figure 5.6b and Figure 5.6c. However, our experimental results reveal that in some cases, relying only on the minimum and maximum RSS values for the device mapping is not a good strategy, as demonstrated in Figure 5.6d. For the Flyer – eeePC device pair, the MM method fails to calibrate the eeePC device and the resulting localization error almost doubles, compared to the SC and SCmed methods. In particular, the MM method computes the fitting parameters (A, B) as $(0.76, -27.35)$, while the optimal values attained by the MC method are $(0.89, -11.09)$. This is due to the fact that the eeePC device recorded an outlier maximum RSS value that caused the B parameter to deviate from the optimal, thus leading to poor calibration and consequently high localization error. Similar behaviour was observed for other device combinations, as shown in the following.

We have also investigated the performance of the calibration methods in case the user-carried device is the same as the device used to collect the RSS data for the radiomap [88]. Even though, in many real life applications the localization system is expected to track mostly diverse devices, there is always the possibility of a user carrying the same device. We observed that the localization error of the SSD and HLF methods is increased compared to the self-calibration method and in fact it would be better not to employ them at all if this situation can be identified (e.g., the device transmits its brand and model to the localization system). This behaviour is due to the smaller dimensionality of the SSD and HLF fingerprints [101]. Moreover, transforming the RSS values in SSD and HLF methods reduces the discriminative capabilities of the RSS values at the expense of better accuracy when heterogeneous devices are considered.

The results for five different user-carried devices assuming that the reference device is either the Asus eeePC or the HTC Desire are summarized in Table 5.2 and Table 5.3, respectively. For every device pair the median of the mean positioning error $\bar{\epsilon}$ in meters is reported for various calibration methods, while each row indicates the device used during localization. The calibration performance of the MM method seems to be affected in several cases, e.g., the eeePC – iPAQ, the eeePC – Nexus S (Table 5.2) or the Desire – eeePC (Table 5.3) device pairs.

Looking at these results it is evident that the proposed self-calibration method improves accuracy for all device pairs and provides similar calibration performance with the MC method. Note that the SCmed method attains the same level of performance with the SC method, despite its reduced computational cost, and interestingly it proves to be more resilient to device heterogeneity; see for example the eeePC – iPAQ (Table 5.2) and the Desire – iPAQ (Table 5.3) device pairs. This is probably because the estimated slope parameter A is usually close to 1, so the intercept parameter B is more important to obtain a good fitting. Thus, in practice, the SCmed method seems to perform better than the SC method for some device pairs because it avoids overfitting when the signal strength histograms have not yet converged.

Table 5.2: Localization error for various calibration methods (eeePC radiomap).

	eeePC							
	SC	**SCmed**	**MM**	**SSD**	**DIFF**	**HLF**	**RBF**	**MC**
iPAQ	3.7	2.8	3.8	4.5	3.4	4.6	5.6	2.7
eeePC	2.2	2.2	2.3	2.7	2.2	2.7	3.6	2.2
Flyer	2.2	2.2	2.7	2.6	2.2	2.5	3.6	2.3
Desire	2.5	2.5	2.5	2.7	2.5	3.0	3.8	2.5
Nexus S	2.3	2.3	2.9	2.9	2.3	2.7	3.9	2.4

We have also investigated the maximum error pertaining to the same 10 testing routes

Table 5.3: Localization error for various calibration methods (Desire radiomap).

	Desire							
	SC	**SCmed**	**MM**	**SSD**	**DIFF**	**HLF**	**RBF**	**MC**
iPAQ	4.2	3.0	3.6	5.5	4.2	5.8	5.1	2.9
eeePC	2.7	2.6	3.6	3.3	2.4	3.3	3.7	2.6
Flyer	2.4	2.3	2.4	3.0	2.3	2.9	3.8	2.4
Desire	2.4	2.3	2.2	2.9	2.3	2.9	3.6	2.3
Nexus S	2.5	2.4	2.7	2.7	2.5	2.7	3.8	2.4

and the results reported in [88] highlight that both self-calibration methods can alleviate high localization errors for most device pairs, while the SCmed method provides a slight improvement over the SC method. Beyond the MC method that is not practical in real-life applications, only the DIFF method achieves similar performance with SCmed. However, as it will be demonstrated in the following, the DIFF method entails high computational cost. Moreover, the SCmed method is still better whenever the iPAQ device is involved.

Thus, according to our findings, the SCmed method is a good candidate solution that exhibits two highly desirable properties, i.e., robustness to device diversity and low computational complexity.

5.4 Differential Fingerprinting

As already discussed, signal strength fingerprinting consists of the *offline* and *online* phases. In the *offline* phase we use a reference device $D^{(1)}$ to collect the reference fingerprints $\mathbf{r}_i = [r_{i1}, \dots, r_{in}]^T$ at the respective locations $\ell_i = (x_i, y_i)$, $i = 1, \dots, l$, which are stored in radiomap $\mathbf{R}^{(1)} \in \mathbb{Z}^-_{n\times l}$. In the *online* phase, we exploit the reference data to estimate location $\widehat{\ell}$, given a new fingerprint $\mathbf{s} = [s_1, \dots, s_n]^T$ measured at the unknown location ℓ by some device $D^{(m)}$, $m = 1, \dots, M$.

In this section, we use the NN method [9] that estimates location by

$$\widehat{\ell}(\mathbf{s}) = \arg\min_{\ell_i} d_i,\ \ d_i^2 = \sum_{j=1}^{n} \left(r_{ij} - s_j\right)^2, \tag{5.8}$$

where d_i^2 is the squared Euclidean distance between the reference fingerprints $\mathbf{r}_i$ and the observed fingerprint $\mathbf{s}$. Essentially, all reference locations are ordered according to increasing d_i and location ℓ_i with the shortest distance between $\mathbf{r}_i$ and $\mathbf{s}$ in the n-dimensional RSS space is returned as the location estimate.

Using signal strength differences removes the constant term K in the *log-distance* propagation model of equation (2.1) and makes the differential fingerprints from diverse devices compatible with each other. This allows the user-carried device to be *any* device $D^{(m')}$, $m' = 1, \dots, M'$ with $M' > M$. There are two approaches in the literature for creating the differential fingerprints, namely the DIFF and SSD methods, which are defined in the following.

5.4.1 Differences of Signal Strength

DIFF Approach

The DIFF approach [40] creates the differential fingerprints by taking the difference between all pairwise AP combinations, thus the transformed fingerprints contain $\binom{n}{2} = \frac{n(n-1)}{2}$ RSS differences. In this sense, the DIFF reference fingerprint $\tilde{\mathbf{r}}_i$ at location ℓ_i in the differential radiomap $\tilde{\mathbf{R}}$ and the DIFF fingerprint during localization $\tilde{\mathbf{s}}$ are defined as

$$\tilde{\mathbf{r}}_i = [\tilde{r}_{i12}, \dots, \tilde{r}_{i(n-1)n}]^T \tag{5.9}$$

$$\tilde{\mathbf{s}} = [\tilde{s}_{12}, \dots, \tilde{s}_{(n-1)n}]^T, \tag{5.10}$$

where $\tilde{r}_{ijk} = r_{ij} - r_{ik}$ and $\tilde{s}_{jk} = s_j - s_k$, $1 \le j < k \le n$ denote the RSS difference between the j-th and k-th APs in the reference and localization fingerprint, respectively. Positioning with the NN method is performed by replacing d_i^2 in equation (5.8) with

$$\tilde{d}_i^2 = \sum_{k=2}^{n} \sum_{j=1}^{k-1} \left(\tilde{r}_{ijk} - \tilde{s}_{jk}\right)^2. \tag{5.11}$$

For convenience, we use the following Lemma that holds for DIFF fingerprints.

Lemma 4.

$$\begin{aligned}\tilde{d}_i^2 &= \sum_{k=2}^{n} \sum_{j=1}^{k-1} \left(\tilde{r}_{ijk} - \tilde{s}_{jk}\right)^2 \\ &= \sum_{k=2}^{n} \sum_{j=1}^{k-1} \left(r_{ij} - r_{ik} - (s_j - s_k)\right)^2 \\ &= \frac{1}{2} \sum_{k=1}^{n} \sum_{j=1}^{n} B_{ijk}^2,\end{aligned} \tag{5.12}$$

where $B_{ijk} = r_{ij} - r_{ik} - (s_j - s_k)$. The complete proof of Lemma 4 using mathematical induction is included in Appendix G.

Now, let $\mathbf{r}_i$ be the RSS reference fingerprint at location ℓ_i and $\mathbf{s}$ the RSS positioning fingerprint in a localization area covered by n APs. As a consequence of the DIFF definition, we prove the following theorem that relates the DIFF with the traditional RSS fingerprints in case the NN method is used for localization.

Theorem 1. *The distance* $\tilde{d}_i^2$ *between the DIFF fingerprints* $\tilde{\mathbf{r}}_i$ *and* $\tilde{\mathbf{s}}$ *with respect to the distance* d_i^2 *between the corresponding RSS fingerprints* $\mathbf{r}_i$ *and* $\mathbf{s}$ *is given by*

$$\tilde{d}_i^2 = n\Big(d_i^2 - n(\bar{r}_i - \bar{s})^2\Big). \tag{5.13}$$

Proof. Starting from Lemma 4 we have for the inner sum

$$\begin{aligned}\sum_{j=1}^{n} B_{ijk}^2 &= \sum_{j=1}^{n} \Big(r_{ij} - r_{ik} - (s_j - s_k)\Big)^2 \\ &= \sum_{j=1}^{n} \Big(r_{ij} - s_j - (r_{ik} - s_k)\Big)^2 \\ &= \sum_{j=1}^{n} (r_{ij} - s_j)^2 - 2\sum_{j=1}^{n} C_{ijk} + \sum_{j=1}^{n} (r_{ik} - s_k)^2 \\ &= d_i^2 - 2\sum_{j=1}^{n} C_{ijk} + n(r_{ik} - s_k)^2, \end{aligned} \tag{5.14}$$

where $C_{ijk} = (r_{ij} - s_j)(r_{ik} - s_k)$ and for the last equality we have used the squared Euclidean distance between RSS fingerprints in equation (5.8). We also have

$$\begin{aligned}\sum_{j=1}^{n} C_{ijk} &= \sum_{j=1}^{n} (r_{ij} - s_j)(r_{ik} - s_k) \\ &= (r_{ik} - s_k)\sum_{j=1}^{n} (r_{ij} - s_j) \\ &= r_{ik}\sum_{j=1}^{n} r_{ij} - r_{ik}\sum_{j=1}^{n} s_j - s_k\sum_{j=1}^{n} r_{ij} + s_k\sum_{j=1}^{n} s_j \\ &= nr_{ik}\bar{r}_i - nr_{ik}\bar{s} - n\bar{r}_i s_k + ns_k\bar{s}. \end{aligned} \tag{5.15}$$

By replacing equations (5.14) and (5.15) into equation (5.12) we get

$$\begin{aligned}\tilde{d}_i^2 &= \frac{1}{2}\sum_{k=1}^{n}\Big(d_i^2 - 2\sum_{j=1}^{n} C_{ijk} + n(r_{ik}-s_k)^2\Big)\\
&= \frac{1}{2}\sum_{k=1}^{n} d_i^2 - \sum_{k=1}^{n}\sum_{j=1}^{n} C_{ijk} + \frac{n}{2}\sum_{k=1}^{n}(r_{ik}-s_k)^2\\
&= \frac{n}{2}d_i^2 - \sum_{k=1}^{n}(nr_{ik}\bar{r}_i - nr_{ik}\bar{s} - n\bar{r}_i s_k + ns_k\bar{s}) + \frac{n}{2}d_i^2\\
&= nd_i^2 - n\bar{r}_i\sum_{k=1}^{n} r_{ik} + n\bar{s}\sum_{k=1}^{n} r_{ik} + n\bar{r}_i\sum_{k=1}^{n} s_k - n\bar{s}\sum_{k=1}^{n} s_k\\
&= nd_i^2 - n^2\bar{r}_i^2 + n^2\bar{r}_i\bar{s} + n^2\bar{r}_i\bar{s} - n^2\bar{s}^2\\
&= n\Big(d_i^2 - n(\bar{r}_i^2 - 2\bar{r}_i\bar{s} + \bar{s}^2)\Big)\\
&= n\Big(d_i^2 - n(\bar{r}_i - \bar{s})^2\Big). \qquad (5.16)\end{aligned}$$

□

SSD Approach

In the SSD method [56, 101], the differential fingerprints are created by subtracting the RSS value of an anchor AP from the other RSS values in the original fingerprint. Thus, the transformed fingerprints contain only the $n-1$ RSS differences that are independent. The anchor AP can be selected as the one that exhibits the least average deviation of RSS values over the whole localization area [56]. Without loss of generality we assume that ρ is the anchor AP and we define the SSD reference fingerprint $\check{\mathbf{r}}_i$ at location ℓ_i in the differential radiomap $\check{\mathbf{R}}$ and the SSD fingerprint during localization $\check{\mathbf{s}}$ as

$$\begin{aligned}\check{\mathbf{r}}_i &= [\check{r}_{i1},\ldots,\check{r}_{i(n-1)}]^T \qquad (5.17)\\
\check{\mathbf{s}} &= [\check{s}_1,\ldots,\check{s}_{n-1}]^T, \qquad (5.18)\end{aligned}$$

where $\check{r}_{ij} = r_{ij} - r_{i\rho}$ and $\check{s}_j = s_j - s_\rho$, $j = 1,\ldots,n,\ j \neq \rho$ denote the RSS difference between the j-th AP and the anchor AP ρ in the reference and localization fingerprint, respectively. Localization with NN is performed by replacing d_i^2 with

$$\check{d}_i^2 = \sum_{\substack{j=1\\ j\neq\rho}}^{n}\left(\check{r}_{ij} - \check{s}_j\right)^2. \qquad (5.19)$$

Similarly with the DIFF approach, we prove the following theorem that relates the SSD with the traditional RSS fingerprints in case the NN method is used for localization.

Theorem 2. *The distance $\check{d}_i^2$ between the SSD fingerprints $\check{\mathbf{r}}_i$ and $\check{\mathbf{s}}$ with respect to the distance d_i^2 between the corresponding RSS fingerprints $\mathbf{r}_i$ and $\mathbf{s}$ is given by*

$$\check{d}_i^2 = d_i^2 - 2n(r_{i\rho} - s_\rho)(\bar{r}_i - \bar{s}) + n(r_{i\rho} - s_\rho)^2, \tag{5.20}$$

where ρ is the reference AP used to create the SSD fingerprints.

Proof. Starting from equation (5.19) we have

$$\begin{aligned}
\check{d}_i^2 &= \sum_{\substack{j=1\\ j\neq\rho}}^{n} \left(\check{r}_{ij} - \check{s}_j\right)^2 \\
&= \sum_{\substack{j=1\\ j\neq\rho}}^{n} \left(r_{ij} - r_{i\rho} - (s_j - s_\rho)\right)^2 \\
&= \sum_{j=1}^{n} \left(r_{ij} - s_j - (r_{i\rho} - s_\rho)\right)^2 \\
&= \sum_{j=1}^{n} (r_{ij} - s_j)^2 - 2\sum_{j=1}^{n} (r_{ij} - s_j)(r_{i\rho} - s_\rho) + \sum_{j=1}^{n} (r_{i\rho} - s_\rho)^2 \\
&= d_i^2 - 2(r_{i\rho} - s_\rho)\Big(\sum_{j=1}^{n} r_{ij} - \sum_{j=1}^{n} s_j\Big) + n(r_{i\rho} - s_\rho)^2 \\
&= d_i^2 - 2n(r_{i\rho} - s_\rho)(\bar{r}_i - \bar{s}) + n(r_{i\rho} - s_\rho)^2.
\end{aligned} \tag{5.21}$$

□

5.4.2 Mean Differential Fingerprints

Looking at equation (5.11) we can see that the DIFF method may increase dramatically the dimensionality of the fingerprints, especially is areas covered by a large number of APs, thus leading to higher computational complexity compared to the traditional RSS fingerprints. On the other hand, equation (5.19) suggests that the dimension of the SSD fingerprints is in the same order as RSS fingerprints. However, selecting an anchor AP is not trivial, especially in large scale setups where the APs provide partial coverage. Moreover, recent experimental studies report that the DIFF method achieves higher localization accuracy than the SSD method [44]. Our experimental results in Section 5.3.3 also confirm this observation.

Our goal is to keep the best of these two methods in order to preserve the localization accuracy, while keeping the computational overhead low. We propose the Mean Differential Fingerprint (MDF) approach that uses the mean value of the RSS fingerprint to create the RSS differences. In the following, we formally define the MDF fingerprints.

For convenience, let $\bar{r}_i = \frac{1}{n}\sum_{j=1}^{n} r_{ij}$ and $\bar{s} = \frac{1}{n}\sum_{j=1}^{n} s_j$ denote the mean RSS value over all APs in the reference RSS fingerprint $\mathbf{r}_i$ and the localization fingerprint $\mathbf{s}$, respectively. The MDF fingerprint $\bar{\bar{\mathbf{r}}}_i$ at location ℓ_i in the differential radiomap $\bar{\bar{\mathbf{R}}}$ and the MDF fingerprint during localization $\bar{\bar{\mathbf{s}}}$ are defined as

$$\bar{\bar{\mathbf{r}}}_i = [\bar{\bar{r}}_{i1}, \ldots, \bar{\bar{r}}_{in}]^T \quad (5.22)$$

$$\bar{\bar{\mathbf{s}}} = [\bar{\bar{s}}_1, \ldots, \bar{\bar{s}}_n]^T, \quad (5.23)$$

where $\bar{\bar{r}}_{ij} = r_{ij} - \bar{r}_i$ and $\bar{\bar{s}}_j = s_j - \bar{s}, \ j = 1, \ldots, n$ denote the RSS difference between the j-th AP and the mean RSS value in the reference and localization fingerprint, respectively. Finally, positioning with the NN method is performed by replacing d_i^2 with

$$\bar{\bar{d}}_i^2 = \sum_{j=1}^{n} \left(\bar{\bar{r}}_{ij} - \bar{\bar{s}}_j\right)^2. \quad (5.24)$$

Note that the MDF fingerprints have the same dimension as the original RSS fingerprints, while from the processing power perspective only the mean RSS value $\bar{s}$ needs to be calculated during localization. Moreover, by observing the radio propagation model in equation (2.1) we can easily see that the MDF approach removes the device-dependent term K, thus addresses the device diversity problem equally well compared to DIFF and SSD methods.

As a consequence of the MDF definition, we can derive the relation between the MDF and the traditional RSS fingerprints in case the NN method is used for positioning. First, we see that the following Lemma holds.

Lemma 5. *For a vector* $\mathbf{x} = [x_1, \ldots, x_n]^T$ *with mean value* $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ *it holds that*

$$\sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2.$$

The proof of Lemma 5 is included in Appendix G. We now prove the following theorem for the MDF fingerprints.

Theorem 3. *The distance* $\bar{\bar{d}}_i^2$ *between the MDF fingerprints* $\bar{\bar{\mathbf{r}}}_i$ *and* $\bar{\bar{\mathbf{s}}}$ *with respect to the distance* d_i^2 *between the corresponding RSS fingerprints* $\mathbf{r}_i$ *and* $\mathbf{s}$ *is given by*

$$\bar{\bar{d}}_i^2 = d_i^2 - n(\bar{r}_i - \bar{s})^2. \quad (5.25)$$

Proof. Starting from equation (5.24) we have

$$\begin{aligned}
\bar{\bar{d}}_i^2 &= \sum_{j=1}^{n} \left(\bar{\bar{r}}_{ij} - \bar{\bar{s}}_j\right)^2 \\
&= \sum_{j=1}^{n} \left(r_{ij} - \bar{r}_i - (s_j - \bar{s})\right)^2 \\
&= \sum_{j=1}^{n} \left((r_{ij} - \bar{r}_i)^2 - 2(r_{ij} - \bar{r}_i)(s_j - \bar{s}) + (s_j - \bar{s})^2\right) \\
&= \sum_{j=1}^{n} (r_{ij} - \bar{r}_i)^2 - 2\sum_{j=1}^{n} A_{ij} + \sum_{j=1}^{n} (s_j - \bar{s})^2 \\
&= \sum_{j=1}^{n} r_{ij}^2 - n\bar{r}_i^2 - 2\sum_{j=1}^{n} A_{ij} + \sum_{j=1}^{n} s_j^2 - n\bar{s}^2, \qquad (5.26)
\end{aligned}$$

where $A_{ij} = (r_{ij} - \bar{r}_i)(s_j - \bar{s})$ and for the last equality we have used Lemma 5. We also have

$$\begin{aligned}
\sum_{j=1}^{n} A_{ij} &= \sum_{j=1}^{n} (r_{ij} - \bar{r}_i)(s_j - \bar{s}) \\
&= \sum_{j=1}^{n} \left(r_{ij}s_j - r_{ij}\bar{s} - \bar{r}_i s_j + \bar{r}_i\bar{s}\right) \\
&= \sum_{j=1}^{n} r_{ij}s_j - \bar{s}\sum_{j=1}^{n} r_{ij} - \bar{r}_i \sum_{j=1}^{n} s_j + n\bar{r}_i\bar{s} \\
&= \sum_{j=1}^{n} r_{ij}s_j - n\bar{r}_i\bar{s} - n\bar{r}_i\bar{s} + n\bar{r}_i\bar{s} \\
&= \sum_{j=1}^{n} r_{ij}s_j - n\bar{r}_i\bar{s}. \qquad (5.27)
\end{aligned}$$

By replacing equation (5.27) into equation (5.26) and using the squared Euclidean distance between RSS fingerprints given by equation (5.8), we get

$$\begin{aligned}
\bar{\bar{d}}_i^2 &= \sum_{j=1}^{n} r_{ij}^2 - n\bar{r}_i^2 - 2\sum_{j=1}^{n} r_{ij}s_j + 2n\bar{r}_i\bar{s} + \sum_{j=1}^{n} s_j^2 - n\bar{s}^2 \\
&= \sum_{j=1}^{n} (r_{ij} - s_j)^2 - n(\bar{r}_i - \bar{s})^2 \\
&= d_i^2 - n(\bar{r}_i - \bar{s})^2. \qquad (5.28)
\end{aligned}$$

□

Combining Theorem 1 and Theorem 3 the following corollary holds.

Corollary 3. *The distance $\bar{\bar{d}}_i^2$ between the MDF fingerprints $\bar{\bar{\mathbf{r}}}_i$ and $\bar{\bar{\mathbf{s}}}$ is proportional to the distance $\tilde{d}_i^2$ between the DIFF fingerprints $\tilde{\mathbf{r}}_i$ and $\tilde{\mathbf{s}}$*

$$\bar{\bar{d}}_i^2 = \frac{1}{n}\tilde{d}_i^2. \tag{5.29}$$

The importance of this result is that, given a RSS fingerprint $\mathbf{s}$ during positioning, the ordering of the candidate locations $\ell_i \in L$ is preserved when either MDF or DIFF fingerprints are used. Thus, the NN localization method provides exactly the same location estimates in both cases. The results presented in the following confirm that the proposed MDF approach achieves the same level of accuracy with DIFF, but with significantly lower computational complexity, due to the lower dimension of the MDF fingerprints.

5.4.3 Probability of Correct Location Estimation

In order to gain more insight about the behaviour of the differential fingerprints, we derive analytical models to determine the probability of returning the correct location during localization.

Let ℓ_1 and ℓ_2 be two neighbouring locations in the localization area, while $\mathbf{r}_1$ and $\mathbf{r}_2$ are the corresponding RSS fingerprints in the radiomap. We assume that during localization the user is located at ℓ_1 and observes the RSS fingerprint $\mathbf{s}$. The fingerprint $\mathbf{s}$ is a normally distributed random vector, i.e. $\mathbf{s} \sim \mathcal{N}(\mathbf{r}_1, \Sigma)$, where $\Sigma = \sigma^2 I_n$ is the covariance matrix, while σ^2 is the variance of the Gaussian noise that disturbs the RSS values and I_n is the identity matrix. The objective is to derive analytically the probability that the NN localization method will return the correct location ℓ_1, instead of the incorrect location ℓ_2 when RSS, DIFF, MDF or SSD fingerprints are used, respectively. Next, we generalize our results for areas where the radiomap contains several location fingerprints.

Analytical Model for RSS Fingerprints

In the case of traditional RSS fingerprints, the NN method will return the correct location ℓ_1 only if the following condition is satisfied

$$d_1^2 \leq d_2^2. \tag{5.30}$$

In other words the unknown user location is correctly identified only if the distance between observed fingerprint $\mathbf{s}$ and the correct location fingerprint $\mathbf{r}_1$ is smaller than the distance between the observed fingerprint and the incorrect neighbouring location fingerprint $\mathbf{r}_2$.

Starting from equation (5.30) we have

$$\begin{aligned} d_1^2 \le d_2^2 \Leftrightarrow & \sum_{j=1}^{n}(r_{1j}-s_j)^2 \le \sum_{j=1}^{n}(r_{2j}-s_j)^2 \\ \Leftrightarrow & \sum_{j=1}^{n} r_{1j}^2 - 2\sum_{j=1}^{n} r_{1j}s_j + \sum_{j=1}^{n} s_j^2 \le \sum_{j=1}^{n} r_{2j}^2 - 2\sum_{j=1}^{n} r_{2j}s_j + \sum_{j=1}^{n} s_j^2 \\ \Leftrightarrow & \, 2\sum_{j=1}^{n}\beta_j s_j + \sum_{j=1}^{n}\gamma_j \le 0, \end{aligned} \tag{5.31}$$

where $\beta_j = (r_{2j} - r_{1j})$ and $\gamma_j = (r_{1j}^2 - r_{2j}^2)$. For convenience, we use vector notation to rewrite equation (5.31) as

$$\boldsymbol{\beta}\mathbf{s} + \gamma \le 0, \tag{5.32}$$

where $\boldsymbol{\beta} = 2[\beta_1, \dots, \beta_n]$ and $\gamma = \sum_{j=1}^{n} \gamma_j$.

The random variable $C = \boldsymbol{\beta}\mathbf{s} + \gamma$ is normally distributed, as a linear function of the multivariate normal vector $\mathbf{s}$, i.e. $C \sim \mathcal{N}(\mu_C, \sigma_C^2)$ with

$$\begin{aligned} \mu_C &= \boldsymbol{\beta}\mathbf{r}_1 + \gamma \\ \sigma_C^2 &= \boldsymbol{\beta}\Sigma\boldsymbol{\beta}^T. \end{aligned} \tag{5.33}$$

This is equivalent with the result reported in [57]. Thus, the probability of correct location estimation, when the NN method compares the observed fingerprint with just two location fingerprints in the radiomap, is given by

$$Pr\{C \le 0\} = \frac{1}{2} + \frac{1}{2}\operatorname{erf}\left(\frac{-\mu_C}{\sqrt{2\sigma_C^2}}\right). \tag{5.34}$$

Finally, in a real localization system where the radiomap is expected to contain several location fingerprints, depending on the size of the area and the density of the reference locations, the probability of correct location estimation $\mathcal{P}_C$ can be calculated as

$$\mathcal{P}_C = Pr\{C_2 \le 0, \dots, C_l \le 0\} \approx \prod_{i=2}^{l} Pr\{C_i \le 0\}, \tag{5.35}$$

where the variable C_i corresponds to the condition $d_1^2 \le d_i^2$, $i = 2, \dots, l$. Although this analytical model is based on the assumption that the random variables C_i are independent, which is not true, it still provides a good approximation of the probability of correct location estimation [57]. We also validate the analytical model in equation (5.35) with simulations later in Section 5.4.5.

Analytical Model for MDF and DIFF Fingerprints

We follow a similar approach to obtain the analytical model for the probability of correct location estimation when MDF, rather than RSS, fingerprints are used in the NN localization method. In this case, using Theorem 3, we can show that

$$\bar{\bar{d}}_1^2 \leq \bar{\bar{d}}_2^2 \Leftrightarrow 2\sum_{j=1}^{n} \beta_j (s_j - \bar{s}) + \sum_{j=1}^{n} \delta_j \leq 0, \tag{5.36}$$

where $\beta_j = (r_{2j} - r_{1j})$ and $\delta_j = (r_{1j}^2 - \bar{r}_1^2 - r_{2j}^2 + \bar{r}_2^2)$.

Using vector notation, equation (5.36) is equivalent to

$$\boldsymbol{\beta} J \mathbf{s} + \delta \leq 0, \tag{5.37}$$

where $\boldsymbol{\beta} = 2[\beta_1, \ldots, \beta_n]$, $J = I - \frac{1}{n}\mathbf{e}^T\mathbf{e}$, $\mathbf{e} = [1, \ldots, 1]$ and $\delta = \sum_{j=1}^{n} \delta_j$. It can be shown that the random variable $Q = \boldsymbol{\beta} J \mathbf{s} + \delta$ is normally distributed, i.e. $Q \sim \mathcal{N}(\mu_Q, \sigma_Q^2)$ with mean and variance

$$\begin{aligned} \mu_Q &= \boldsymbol{\beta} J \mathbf{r}_a + \delta \\ \sigma_Q^2 &= \boldsymbol{\beta} J \Sigma J^T \boldsymbol{\beta}^T. \end{aligned} \tag{5.38}$$

The detailed derivation of the inequality (5.36) and the proof that Q is a normally distributed random variable is included in Appendix H.1.

Consequently, the probability of correct location estimation, using the NN localization method with MDF fingerprints, in the two-location case and the multiple location setup respectively, are given by

$$Pr\{Q \leq 0\} = \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{-\mu_Q}{\sqrt{2\sigma_Q^2}}\right) \tag{5.39}$$

$$\mathcal{P}_Q \approx \prod_{i=2}^{l} Pr\{Q_i \leq 0\}. \tag{5.40}$$

It follows directly from Corollary 3 that the analytical model for the DIFF approach, is exactly the same with the model derived for the MDF approach.

Analytical Model for SSD Fingerprints

When SSD fingerprints are considered, using Theorem 2 with ρ being the anchor AP, we can show that

$$\check{d}_1^2 \leq \check{d}_2^2 \Leftrightarrow 2\sum_{j=1}^{n} (\beta_j - \beta_\rho)(s_j - s_\rho) + \sum_{j=1}^{n} \epsilon_j \leq 0, \tag{5.41}$$

where $\beta_j = (r_{2j} - r_{1j})$, $\beta_\rho = (r_{2\rho} - r_{1\rho})$ and $\epsilon_j = (r_{1j}^2 + r_{1\rho}^2 - r_{2j}^2 - r_{2\rho}^2 - 2r_{1\rho}\bar{r}_1 + 2r_{2\rho}\bar{r}_2)$. Again, using vector notation we may rewrite equation 5.41 as

$$\boldsymbol{\eta} M \mathbf{s} + \epsilon \leq 0, \tag{5.42}$$

where $\boldsymbol{\eta} = 2[\beta_1 - \beta_\rho, \ldots, \beta_n - \beta_\rho]$, $M = I - \mathbf{e}^T\mathbf{u}$, $\mathbf{e} = [1, \ldots, 1]$, $\mathbf{u} = [0, \ldots, 1, \ldots, 0]$ and $\epsilon = \sum_{j=1}^{n} \epsilon_j$. The detailed derivation is relegated to Appendix H.2.

Following the same process with the MDF approach, we can show that the random variable $R = \boldsymbol{\eta} M \mathbf{s} + \epsilon$ is normally distributed, i.e. $R \sim \mathcal{N}(\mu_R, \sigma_R^2)$ with mean and variance

$$\begin{aligned} \mu_R &= \boldsymbol{\eta} M \mathbf{r}_1 + \epsilon \\ \sigma_R^2 &= \boldsymbol{\eta} M \Sigma M^T \boldsymbol{\eta}^T. \end{aligned} \tag{5.43}$$

Finally, the probabilities that the NN localization method returns the correct location when it compares the observed SSD fingerprint with just two SSD location fingerprints and several SSD location fingerprints respectively are given by

$$Pr\{R \leq 0\} = \frac{1}{2} + \frac{1}{2}\text{erf}\left(\frac{-\mu_R}{\sqrt{2\sigma_R^2}}\right) \tag{5.44}$$

$$\mathcal{P}_R \approx \prod_{i=2}^{l} Pr\{R_i \leq 0\}. \tag{5.45}$$

5.4.4 Performance Analysis of Differential Fingerprinting

We now use the analytical models for RSS given by equation (5.35), for MDF and DIFF given by equation (5.40) and for SSD given by equation (5.45) to analyse the performance of the differential fingerprinting approaches.

We consider a simple localization setup, adopted from [57], which is depicted in Figure 5.7. The numbered WLAN APs (marked with triangles) are deployed in the perimeter of the localization area and we start with $n = 3$ APs that are placed at the corners of the area, while the maximum number is $n = 16$ APs. There are $l = 9$ reference locations (marked with circles) that are uniformly spread over a square grid, while the grid spacing is 1 m.

We assume that the RSS values from the surrounding APs at the reference locations are collected with device $D^{(1)}$ and are given by the log-distance propagation model

$$r_{ij}^{(1)} = K - 10f\log_{10} d_{ij},\ i = 1, \ldots, l,\ j = 1, \ldots, n, \tag{5.46}$$

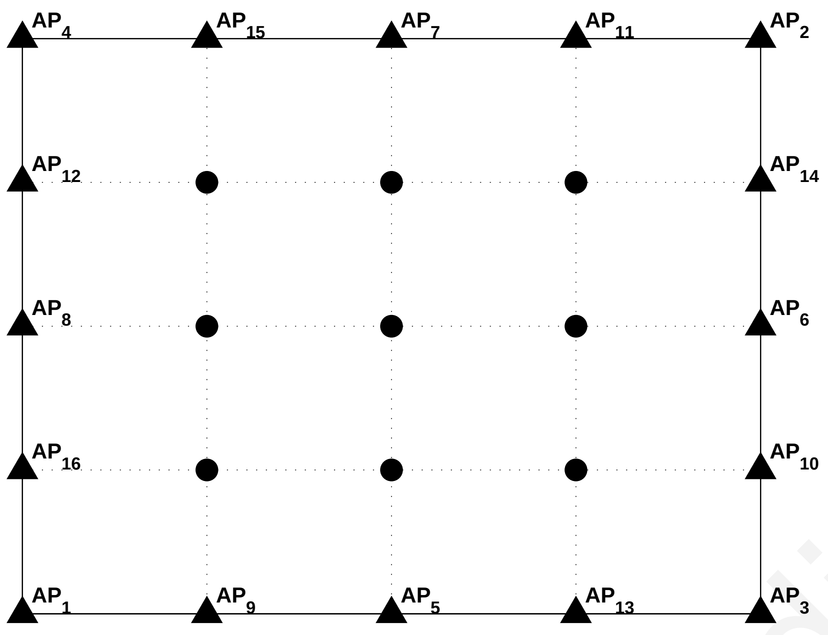


Figure 5.7: Small-scale localization setup.

where d_{ij} denotes the distance between the i-th reference location and the j-th AP. We use typical values for the model parameters and set $K = -22.7$ dBm, while $f = 3.3$. Moreover, we assume that $r_{ij}^{(1)}$ is deterministic because it has been averaged over a sufficiently large number of samples to filter out the noise.

During localization, the user is located at the central location $\ell_1 = (2, 2)$ and we assume that the user carries either the same device $D^{(1)}$ or a different device $D^{(2)}$. When $D^{(1)}$ is the user-carried device, the observed fingerprint is $\mathbf{s}^{(1)} \sim \mathcal{N}(\mathbf{r}_1^{(1)}, \Sigma)$, where $\Sigma = \sigma^2 I$. When the second device $D^{(2)}$ is carried, we assume a linear relation between the RSS values reported by $D^{(2)}$ and the corresponding RSS values of $D^{(1)}$, such that

$$\mathbf{s}^{(2)} \sim \mathcal{N}(A_{12}\mathbf{r}_1^{(1)} + B_{12}, \Sigma), \tag{5.47}$$

where (A_{12}, B_{12}) are the fitting parameters between devices $D^{(2)}$ and $D^{(1)}$. Such linear relation between RSS values measured by heterogeneous devices has been reported in several experimental studies [52, 88, 116].

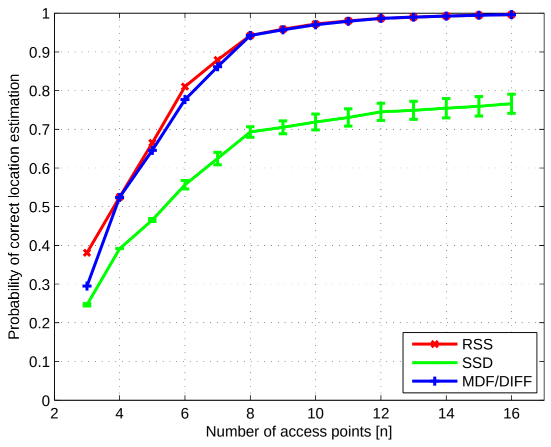
We start out with the case of a user carrying the same device $D^{(1)}$ that was used to collect the fingerprints in the radiomap. The results for varying number of APs, assuming that the standard deviation of the noise disturbing the RSS values is $\sigma = 3$ dBm, are illustrated in Figure 5.8a. We observe that using MDF or DIFF fingerprints does not have a considerable effect and the performance is quite similar with the case of using the traditional RSS fingerprints. The probability of correct location estimation $\mathcal{P}_Q$ is around 80% when 6 APs are considered and seems to converge to 100% as the number of APs increases. On the other hand, performance is significantly degraded when SSD fingerprints are used and interestingly $\mathcal{P}_R < 80\%$ (mean value), even when all 16 APs are considered. The errorbars in the SSD curve reflect the standard deviation with respect to different anchor AP ρ. The result in Figure 5.8a implies that the selection of the anchor AP is not critical and the SSD approach has reduced accuracy, even if the user carries the same device that was employed to collect the fingerprints for the radiomap.

The probability of correct location estimation for varying noise standard deviation σ is plotted in Figure 5.8b, where we have fixed the number of APs to $n = 7$ (i.e., only $\text{AP}_j,\ j = 1, \dots, 7$ are considered). We observe that in low noise conditions (e.g., $\sigma = 1$ dBm) the performance of the SSD fingerprints is close to the MDF and DIFF fingerprints. As σ is increased, however, the SSD fingerprints perform worse and $\mathcal{P}_R$ decreases by around 3%–25% depending on the noise level. On the other hand, the MDF and DIFF fingerprints attain almost the same level of performance with the RSS fingerprints.

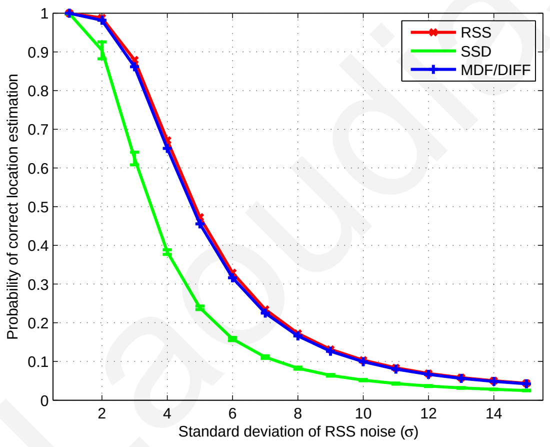
Next, we focus on the case that the user carries a heterogeneous device $D^{(2)}$, which is more likely to happen in real-life applications. For simplicity, we consider a linear relation between the RSS values reported by devices $D^{(1)}$ and $D^{(2)}$, as shown in equation (5.47), and we have set the linear fitting parameters to $(A_{12}, B_{12}) = (0.95, 10)$. In this case, the corresponding probability curves are plotted in Figure 5.8c for increasing number of APs, while $\sigma = 3$ dBm. It is evident that the MDF and DIFF fingerprints are capable of mitigating the device diversity problem, thus maintaining the same level of performance during localization, as in the case of carrying the same device $D^{(1)}$. The SSD fingerprints are also not affected, however they fail to achieve the same level of performance with the MDF and DIFF fingerprints.

As expected, the performance of the RSS fingerprints is poor, especially when a few APs are considered. For instance, $\mathcal{P}_C = 5\%$ when $n = 5$ APs, while $\mathcal{P}_C = 38\%$ when $n = 7$ APs. Their performance, however, is greatly improved when $n \geq 11$ APs are considered and interestingly they outperform the SSD fingerprints. This suggests that in localization areas covered by a large number of APs, the RSS fingerprints may be able to handle different devices and the use of SSD fingerprints might not be justified. Another interesting observation is that the probability curve of the RSS fingerprints peaks at certain points. This situation occurs in case the available APs are uniformly distributed around the localization area, i.e. $n \in \{4, 8, 12, 16\}$ APs, as shown in Figure 5.7. This scenario creates symmetries capable of masking the deficiency of the RSS fingerprints. This observation was also reported in [138], where the authors used a simple 1-D analytical model to explain the behaviour of RSS for diverse devices in symmetrical AP deployments under the assumption that $A_{12} = 1$ and $B_{12} \neq 0$. In contrast, our analytical models apply to the more general 2-D case and $\forall A_{12},\ B_{12}$.

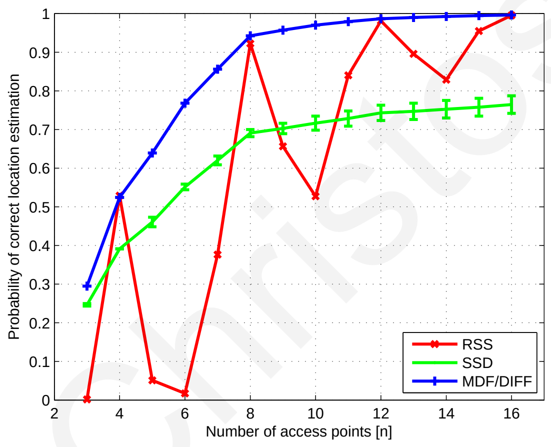
In real-life applications the APs are not expected to be symmetrically deployed and in addition the signal propagation indoors is more complicated compared to our simple localization setup. Moreover, we observe that if RSS fingerprints are used, then $\mathcal{P}_C < 50\%$ when $n = 7$ APs are considered even under low noise conditions; see Figure 5.8d. This implies that the localization accuracy is expected to deteriorate significantly in practice, when RSS fingerprints are used to localize a different device. Therefore, differential fingerprints, specifically using the MDF approach, should be preferred.



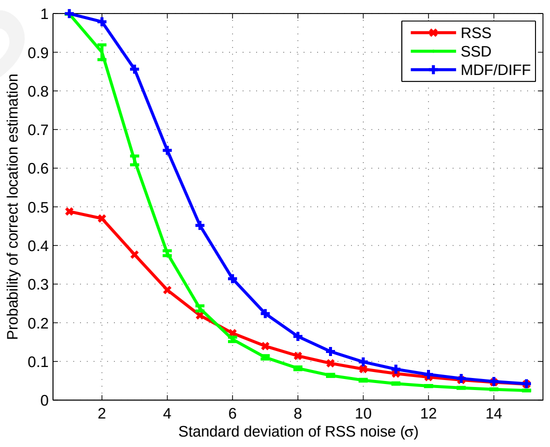
(a) $D^{(1)}$ with varying number of APs ($\sigma = 3$).



(b) $D^{(1)}$ with varying noise (7 APs).



(c) $D^{(2)}$ with varying number of APs ($\sigma = 3$).



(d) $D^{(2)}$ with varying noise (7 APs).

Figure 5.8: Analytical results for differential fingerprinting.

5.4.5 Evaluation of Differential Fingerprinting with Simulations

Next, we validate our analytical results with simulations. We consider the same setup shown in Figure 5.7, while the user resides at location $\ell_1 = (2, 2)$ during localization carrying either the same device $D^{(1)}$ or a different device $D^{(2)}$.

When $D^{(1)}$ is the user-carried device, we generate the RSS localization fingerprint $\mathbf{s}^{(1)}$ by taking the corresponding RSS location fingerprint $\mathbf{r}_1^{(1)}$ and disturbing it with additive Gaussian noise, i.e. $s_j^{(1)} = r_{1j}^{(1)} + X_1$ where $X_1 \sim \mathcal{N}(0, \sigma^2)$. When the second device $D^{(2)}$ is carried, we assume a linear relation to generate the RSS values reported by $D^{(2)}$ such that

$$s_j^{(2)} = A_{12} r_{1j}^{(1)} + B_{12} + X_2, \tag{5.48}$$

where $(A_{12}, B_{12}) = (0.95, 10)$ are the fitting parameters between devices $D^{(2)}$ and $D^{(1)}$, while $X_2 \sim \mathcal{N}(0, \sigma^2)$.

The NN method estimates the user location through the ordering of the 9 candidate locations with respect to increasing Euclidean distance between the localization fingerprint and each location fingerprint in the radiomap using RSS, DIFF, MDF and SSD fingerprints, respectively. In this sense, the probability of correct location estimation can be calculated as

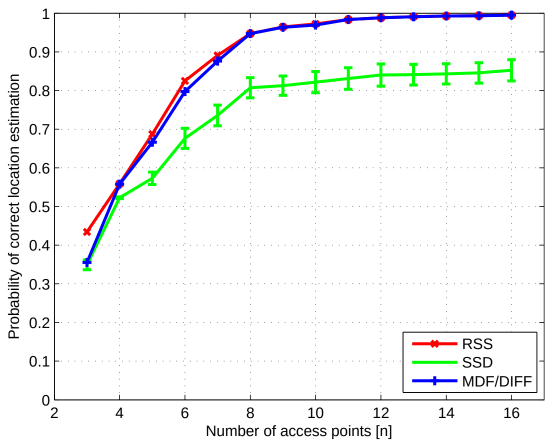
$$\mathcal{P}_Y^{sim} = \frac{N_Y}{N_S}, \; Y \in \{C, Q, R\}, \tag{5.49}$$

where N_Y denotes the number of times that the correct location ℓ_1 was identified using RSS, MDF/DIFF or SSD, while N_S is the total number of simulations. The results pertaining to 10,000 simulated localization fingerprints are illustrated in Figure 5.9.

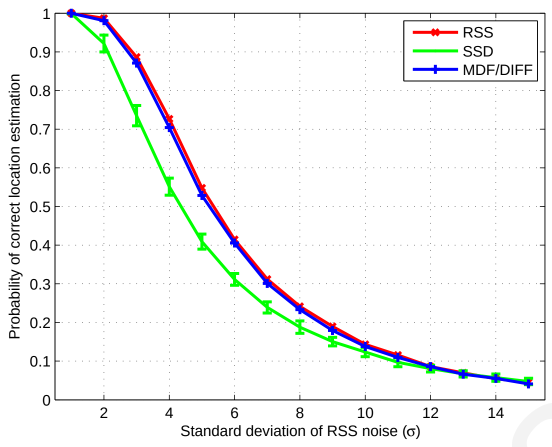
We observe that the analytical models tend to underestimate the probability of correct location estimation compared to the simulation results. For instance, as the number of APs is increased, the $\mathcal{P}_R$ curve for SSD converges to 77% when the analytical model in equation (5.45) is used (recall Figure 5.8a), while the simulation results indicate that $\mathcal{P}_R^{sim}$ converges to 85% as shown in Figure 5.9a. This is expected because in the derivation of the analytical models we assumed that the probabilities of correct location estimation between the correct location and each of the incorrect neighbouring locations are independent. In general, however, we see that the analytical models provide a good approximation and the analytical results reveal similar trends with the simulations for varying number of APs and varying RSS noise both for the same device $D^{(1)}$ (Figure 5.9a and Figure 5.9b), as well for a heterogeneous device $D^{(2)}$ (Figure 5.9c and Figure 5.9d).

5.4.6 Experimental Evaluation of Differential Fingerprinting

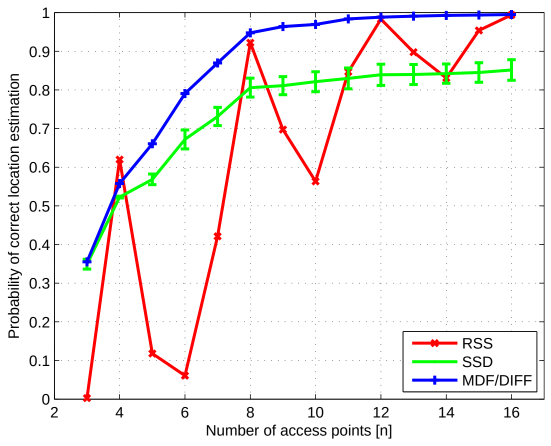
We validate our simulation results using experimental data from the publicly available *KIOS-B* dataset, described in Section 5.1.2. We employ the reference data to build the device-specific radiomaps, by calculating the mean value fingerprint at each reference location. We also create the corresponding DIFF, SSD and MDF radiomaps. Moreover, we use the test data



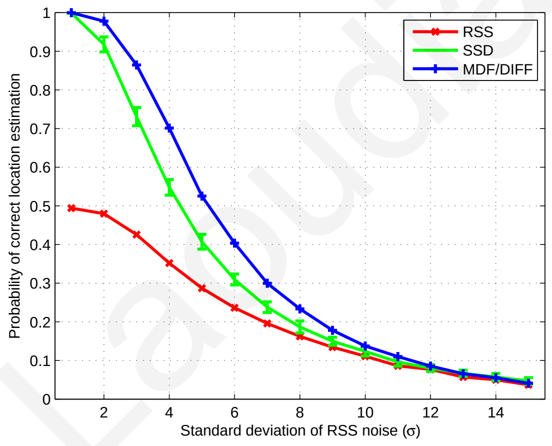
(a) $D^{(1)}$ with varying number of APs ($\sigma = 3$).



(b) $D^{(1)}$ with varying noise (7 APs).



(c) $D^{(2)}$ with varying number of APs ($\sigma = 3$).



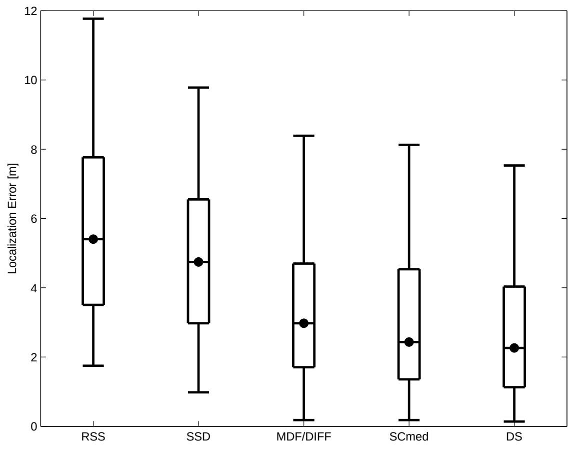
(d) $D^{(2)}$ with varying noise (7 APs).

Figure 5.9: Simulation results for differential fingerprinting.

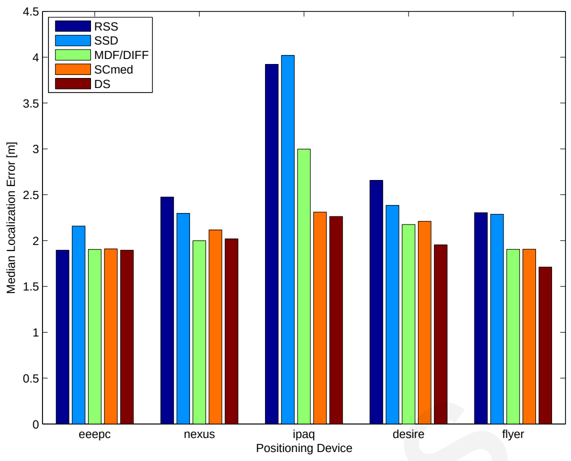
to evaluate the differential fingerprinting approaches in terms of the localization error, which is defined as the physical distance between the actual and estimated user locations. In particular, we examine the distribution of the localization error pertaining to the test data for each device.

The localization error when the Nexus S device was used to create the radiomap, while the user-carried device is the iPAQ, is depicted in Figure 5.10a. The error distribution is shown with box plots, where the central mark indicates the median error, the box edges correspond to the 25th and 75th percentiles and the whiskers extend to the 5th and 95th percentiles of the distribution, respectively. We also consider the case where the RSS radiomap was collected with iPAQ, instead of Nexus S, denoted DS (device-specific). As the radiomap and user-carried devices are the same, DS is the best case that provides the lower bound on the localization error. The SCmed self-calibration method, presented in Section 5.3.2, is also plotted for comparison.

Our first observation is that the localization error is reduced significantly with the pro-



(a) Nexus S radiomap – HP iPAQ user.



(b) Asus eeePC radiomap and different user-carried devices.

Figure 5.10: Localization error of differential approaches for various device pairs.

posed MDF approach. For example, the median error is 3 m when the MDF or DIFF fingerprints are used, compared to 5.4 m with the traditional RSS fingerprints, while the median error is improved less with the SSD fingerprints (4.7 m). Moreover, the MDF fingerprints seem to mitigate higher localization errors that occur when RSS fingerprints are used. In particular, the 75th and 95th percentiles drop from 7.8 m and 11.8 m to 4.7 m and 8.4 m, respectively. More importantly, the performance of MDF is very close to the DS approach that uses the iPAQ radiomap.

The median localization error when the Asus eeePC radiomap is used, while the user carries various devices during localization, is illustrated in Figure 5.10b. We observe that when the localization device is the same with the reference device (Asus eeePC), then the SSD method performs worse than our MDF method, which confirms our analytical and simulation results. The experimental results indicate that the MDF fingerprints attain higher localization accuracy than the RSS fingerprints for all diverse device combinations. Moreover, MDF performs better than SSD in all cases and the median localization error decreases between 8%–25% for different device pairs. Compared to the SCmed device self-calibration method, MDF has similar or marginally better performance, except for the case where the iPAQ device is used for localization.

Next, to investigate the computation time of the differential fingerprinting approaches, we use a Matlab implementation on an Intel Xeon E520 dual core processor 2.4 GHz and 8 GB RAM. The Nexus S–iPAQ device pair is used as before and we consider all 44 APs that are sensed by both devices. The execution times are averaged over 100 runs using the test data from the iPAQ device and the execution times are tabulated in Table 5.4. One location estimate takes around 24 ms using the MDF approach. This is comparable to RSS fingerprinting (23 ms) and as expected the SSD approach requires the same computation time as MDF. On the other hand, the DIFF approach needs 30 ms to estimate the user location, which is 25% higher compared to MDF.

Table 5.4: Computation time [ms] of differential fingerprinting

	RSS	MDF	DIFF	SSD
Time[*]	23	24	30	24

[*]For a single location calculation.

To summarize, the MDF approach should be preferred among the differential fingerprinting methods, because it maintains high level of localization accuracy for different user-carried devices, while the computational overhead remains low. In comparison with the SCmed self-calibration method, MDF is easier to apply in practical localization scenarios. For example, SCmed relies on the buffer size b_s that affects the number of samples required to build the eCDF of the user-carried device for the determination of the linear fitting parameters. The parameter b_s needs to be selected experimentally, while the users still need to walk around for a few seconds until their devices are calibrated. Moreover, if the user takes only a very short walk or moves within a region where the sensed APs have a different distribution than the entire area, then the collected samples might not reflect the eCDF adequately. This may lead to incorrect device calibration and consequently high localization errors. These facts might degrade the performance of the SCmed method in some scenarios.

On the other hand, MDF can be applied in a more straightforward manner, while no parameters need to be fine-tuned. Moreover, the MDF approach facilitates the crowdsourcing of the radiomap with diverse devices by fusing directly the contributions (i.e., RSS differences) of the participating devices, as discussed in the following.

5.5 Crowdsourcing with Differential Fingerprints

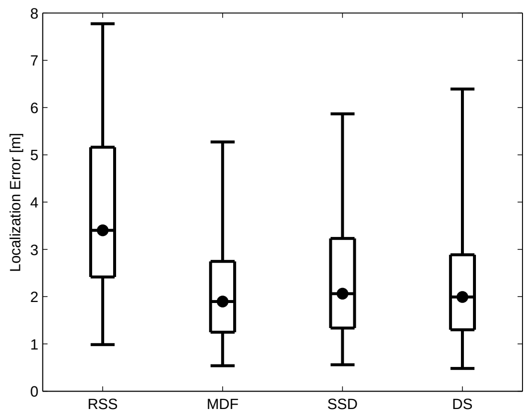
As mentioned in Section 5.2, mobile devices do not report the RSS values in the same way and there is a linear relation between the RSS values measured by heterogeneous devices; see Figure 5.2. Therefore, direct fusion of the RSS radiomaps $\mathbf{R}^{(m)}$ collected with m different devices using equation (5.1) may compromise the quality of the resulting crowdsourced radiomap. To address this issue, we propose to use signal strength differences in order to remove the constant term K in the propagation model given by equation (2.1) and make the fingerprints from diverse devices compatible with each other. In the following, we investigate the differential fingerprinting approaches presented in Section 5.4 for crowdsourcing the radiomap. In particular, we assess the performance of the crowdsourced differential radiomap, using either the MDF, DIFF or the SSD approach, with experimental data from the *KIOS-B* dataset.

5.5.1 Evaluation of Differential Crowdsourcing Approaches

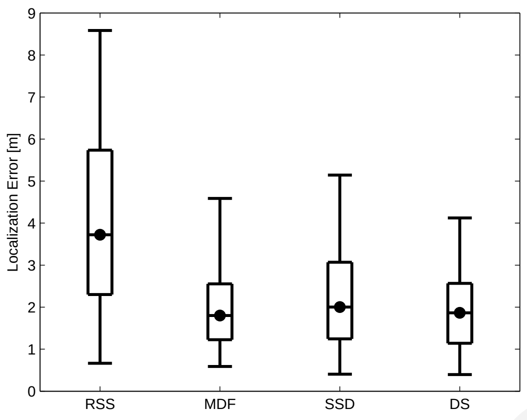
We employ the reference data in the *KIOS-B* dataset to build the device-specific radiomaps and also create the crowdsourced radiomap using different device combinations. Moreover, we use the test data to evaluate the various crowdsourcing approaches in terms of the localization error, which is defined as the physical distance between the actual and estimated user locations, and we examine the distribution of the localization error pertaining to the test set for each device.

In particular, we examine the performance when the RSS crowdsourced radiomap is used, while the distance between the RSS fingerprints in the computations of the NN localization method is given by equation (5.8). We also compare the two variants of the differential crowdsourced radiomap, namely the MDF and SSD approaches where the distance between the fingerprints is given by equations (5.11) and (5.19), respectively. The DIFF approach is omitted because its performance is exactly the same with our MDF approach, but is more computationally expensive as discussed previously. For completeness we report the localization error when the device-specific RSS radiomap of the test device, instead of the crowdsourced RSS radiomap, is considered. This approach, denoted DS, provides the lower bound on the localization error.

First, we consider only two contributing devices, namely the iPAQ and Nexus devices, and each device fully covers the localization area for crowdsourcing the radiomap. The experimental results are depicted in Figure 5.11 with box plots. Figure 5.11a plots the experimental results when the iPAQ serves as the test device for localization. We observe that the localization error is considerably reduced with the differential approaches. For instance, the median error is 3.4 m for the traditional RSS approach, compared with around 2 m for the MDF and SSD approaches. Moreover, the 75th percentile drops from 5.2 m to 2.7 m for MDF and 3.2 m for SSD. More importantly, the performance is very close to the non-crowdsourcing DS approach that uses the RSS radiomap collected only with the iPAQ device. Using another test device (i.e., HTC Desire), which was not considered for crowsourcing, produces similar re-



(a) Localization of the iPAQ device.



(b) Localization of the Desire device.

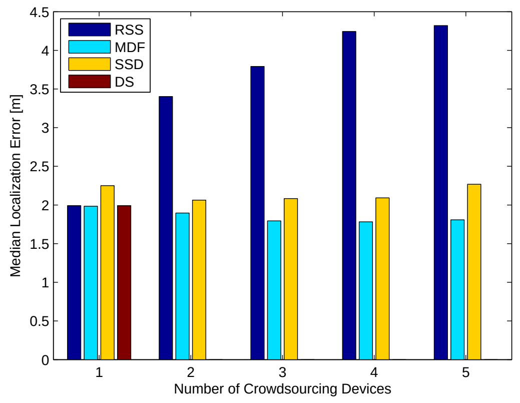
Figure 5.11: Localization with two-device (iPAQ, Nexus) crowdsourced radiomaps.

sults and again the MDF approach filters out the high errors more effectively, compared to SSD; see Figure 5.11b.

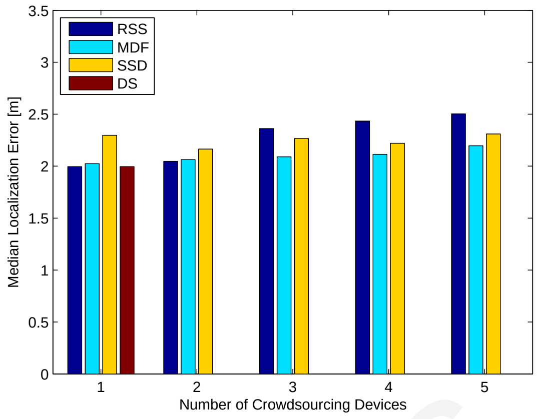
In case more devices are used for crowdsourcing, the performance of the traditional RSS fingerprints may deteriorate further, as shown in Figure 5.12, where the bars depict the median localization error. For instance, using data from five devices to crowdsource the radiomap increases the median localization error to 4.3 m for the iPAQ device (Figure 5.12a). This is much higher compared with 1.8 m median error of the MDF approach and 2.3 m of the SSD approach. We observe that the localization error of both differential approaches does not vary significantly as the number of crowdsourcing devices increases.

Another interesting observation is that the MDF approach seems to perform better than the DS approach. This is not surprising because the area is covered by all five devices, thus the crowdsourced differential radiomap has been created by aggregating data from more than one device in each location, contrary to DS that uses RSS data collected only with the iPAQ device. However, this is not the case with SSD, which also performs worse compared with MDF for any number of contributing devices. This is in line with our experimental results presented earlier in Section 5.4.6. Even though the performance of the RSS approach with respect to the median error can be adequate in some cases, the differential fingerprinting approaches still provide some improvement, as shown in Figure 5.12b, where the eeePC device is localized using non-overlapping radiomaps from a varying number of devices.

The experimental results indicate that the differential approaches are more robust to device diversity and should be preferred for crowdsourcing, especially as the number of the devices that contribute data to the system grows larger. The MDF approach, in particular, brings the full benefit of crowdsourcing regardless of the user-carried device during localization, while



(a) Localization of the iPAQ device with fully overlapping radiomaps.



(b) Localization of the eeePC device with non-overlapping radiomaps.

Figure 5.12: Crowdsourcing with increasing number of contributing devices.

the computational overhead is similar with the case of using traditional RSS fingerprints.

5.6 Chapter Summary

Device diversity is one of the reasons that hinders the proliferation of RSS-based fingerprinting systems. For instance, in traditional fingerprinting systems, where a single device is used to populate the RSS data in the radiomap, the user needs to carry the *same* device during localization to guarantee the best accuracy. Using a different device is feasible, but the RSS values are not usually compatible with the radiomap, leading to accuracy degradation. In fact, heterogeneous mobile devices may report RSS values from the surrounding APs quite differently, even if they are placed at the same location. This is also a major limitation for the emerging crowdsourced fingerprinting systems, where device heterogeneity is inherent due to the diverse mobile devices carried by the contributors.

To this end, we investigate the device diversity issue in fingerprinting systems and revisit the *manual calibration* approach to provide insight on the amount of RSS data that need to be collected at *known* locations with different devices, so that adequate localization accuracy is achieved for heterogeneous devices. Furthermore, we propose a new method based on RSS histograms that runs concurrently with localization and enables a mobile device to be self-calibrated in a short time, thus improving the localization accuracy on-the-fly. Moreover, no user involvement is required in the calibration phase (e.g., visiting several locations and pressing a calibration button on the device) and the tedious data collection is avoided.

Importantly, we develop a novel calibration-free method for diverse devices based on RSS differences. Our approach achieves the same level of localization accuracy with existing differential fingerprinting approaches, but is considerably less expensive in terms of computation time. Our calibration-free approach performs equally well with our self-calibration approach with respect to accuracy, while it exhibits higher applicability in real-life applications. This is because it does not require the fine-tuning of specific parameters, contrary to the self-calibration approach. In addition, our differential fingerprinting approach can be easily extended to crowdsourced localization systems for exploiting the data uploaded by diverse devices in a straightforward way.

Chapter 6

Conclusions

6.1 Summary of Contributions

Our work in the context of localization and tracking in wireless networks contributes to three areas, namely fault tolerant localization and tracking in WSNs, fault tolerant localization in WLAN fingerprinting systems and localization with heterogeneous mobile devices in WLAN fingerprinting systems.

With respect to fault tolerant localization and tracking in binary WSNs our contributions are summarized as follows:

- A novel MC fault model that captures the spatiotemporal dynamics of sensor node faults and is capable of simulating various types of faults [83, 84].
- Formulation of the sensor health state estimation problem as a HMM and derivation of efficient stochastic estimators to infer the unknown sensor states simultaneously with target tracking [76, 84].
- The ftTRACK tracking solution for a single target that combines sensor health state estimation, with target localization and location smoothing by means of Bayesian filtering in centralized WSN architectures [76].
- A fault tolerant multiple target identification approach based on the D-FTLEP leader election protocol and the dSNAP fault tolerant target localization algorithm applicable to distributed WSN architectures [104, 107].

Regarding fault tolerant localization in WLAN fingerprinting systems our contributions include:

- Realistic fault models that capture the effect of WLAN AP malfunctions or adversary attacks [74].
- Robust fault indicators that are either RSS distance-based [85] or likelihood-based, and evaluation of their AP fault detection accuracy and reliability under various fault models [86].
- A class of fault tolerant localization algorithms for WLAN fingerprinting systems that are inspired by the SNAP fault tolerant algorithm applicable in binary WSNs [75] or developed by combining our fault detection mechanisms with modified distance and likelihood-based metrics into hybrid fingerprinting algorithms [86].

Finally, we make the following contributions in regards to localization with heterogeneous mobile devices in WLAN fingerprinting systems:

- An innovative device self-calibration method that uses histograms of RSS values to fit a linear mapping between the device that was used to create the RSS radiomap and the heterogeneous user-carried device [87, 88].
- A novel calibration-free approach based on RSS differences that performs considerably better than existing differential fingerprinting approaches, in terms of localization accuracy and computational complexity.
- Formulation of crowdsourcing in WLAN fingerprinting systems with the introduction of RSS differences and evaluation of various differential fingerprinting approaches for fusing RSS data from heterogeneous devices into a single radiomap [89].

6.2 Directions for Future Work

As part of our future work in fault tolerant localization and tracking in WSNs we plan to investigate and address the following challenges:

- Fault identification and isolation in the context of our ftTRACK target tracking architecture in order to identify the exact type of fault and pinpoint the faulty sensor(s).
- Our sensor health state estimators determine the state of each sensor independently from each other using only the error signal of the individual sensor. Taking into account the spatial correlation of sensor faults in the design of future sensor state estimators is expected to further improve the performance of our sensor health state estimators.

- Distributed sensor health state estimation using information from each sensor's neighbourhood for enabling of a fully distributed ftTRACK variant capable of tracking multiple targets reliably in the presence of sensor faults.

Regarding fault tolerant localization in WLAN fingerprinting systems we plan to pursue the following directions:

- Develop fault identification mechanisms (i.e, recognize the type of fault that has occurred) to trigger the most effective fault tolerant localization algorithm and fault isolation methodologies (i.e, pinpoint the faulty or attacked WLAN AP) for guiding the security or maintenance personnel.

- Apply the analytical modelling approach, which was developed for the device diversity problem in WLAN, to gain more insight into the performance of our fault tolerant algorithms in the presence of different types of faults.

Finally, with respect to localization using heterogeneous mobile devices in WLAN fingerprinting systems our future steps include:

- Integration of the proposed differential fingerprinting approach in our *Anyplace* localization system for commercial Android smartphones [118] to measure and report the savings on computation time and power consumption on real mobile devices, compared with other differential fingerprinting approaches.

Bibliography

[1] U. Ahmad, A. Gavrilov, Y.-K. Lee, and S. Lee, "Context-aware, self-scaling fuzzy ArtMap for received signal strength based location systems," *Soft Computing*, vol. 12, no. 7, pp. 699–713, 2008.

[2] U. Ahmed, A. Gavrilov, S. Lee, and Y.-K. Lee, "Context-aware fuzzy ArtMap for received signal strength based location systems," in *International Joint Conference on Neural Networks (IJCNN)*, 2007, pp. 2740–2745.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, pp. 102–114, Aug. 2002.

[4] M. Alakhras, M. Hussein, and M. Oussalah, "Multivariable fuzzy inference with multi nearest neighbour for indoor WLAN localization based on RSS fingerprint," in *15th International Conference on Computer Modelling and Simulation (UKSim)*, 2013, pp. 656–662.

[5] D. Ampeliotis and K. Berberidis, "Low complexity multiple acoustic source localization in sensor networks based on energy measurements," *Signal Processing*, vol. 90, no. 4, pp. 1300–1312, 2010.

[6] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, "A line in the sand: a wireless sensor network for target detection, classification, and tracking," *Computer Networks*, vol. 46, no. 5, pp. 605–634, Dec. 2004.

[7] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," in *ACM International conference on Embedded networked sensor systems*, 2003, pp. 150–161.

[8] J. J. Astrain, J. Villadangos, J. R. Garitagoitia, J. R. G. de Mendívil, and V. Cholvi, "Fuzzy location and tracking on wireless networks," in *4th ACM international workshop on Mobility management and wireless access (MobiWac)*, 2006, pp. 84–91.

[9] P. Bahl and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *IEEE International Conference on Computer Communications INFOCOM*, vol. 2, 2000, pp. 775–784.

[10] P. Bahl, V. Padmanabhan, and A. Balachandran, "Enhancements to the RADAR user location and tracking system," *Microsoft Research, Tech. Rep. MSR-TR-00-12*, February 2000.

[11] A. Barry, B. Fisher, and M. L. Chang, "A long-duration study of user-trained 802.11 localization," in *2nd international conference on Mobile Entity Localization and Tracking in GPS-less environments (MELT)*, 2009, pp. 197–212.

[12] R. Battiti, T. Nhat, and A. Villani, "Location-aware computing: a neural network model for determining location in wireless LANs," *University of Trento, Trento, Italy, Tech. Rep. DIT-02-0083*, 2002.

[13] C. Beder and M. Klepal, "Fingerprinting based localisation revisited: a rigorous approach for comparing RSSI measurements coping with missed access points and differing antenna attenuations," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2012, pp. 1–7.

[14] E. Bhasker, S. Brown, and W. Griswold, "Employing user feedback for fast, accurate, low-maintenance geolocationing," in *IEEE Conference on Pervasive Computing and Communications (PerCom)*, 2004, pp. 111–120.

[15] P. Bolliger, "Redpin - adaptive, zero-configuration indoor localization through user collaboration," in *1st ACM international workshop on Mobile Entity Localization and Tracking in GPS-less environments (MELT)*, 2008, pp. 55–60.

[16] M. Borenovic, A. Neskovic, D. Budimir, and L. Zezelj, "Utilizing artificial neural networks for WLAN positioning," in *19th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2008, pp. 1–5.

[17] M. Brunato and R. Battiti, "Statistical learning theory for location fingerprinting in wireless LANs," *Computer Networks*, vol. 47, no. 6, pp. 825–845, Apr. 2005.

[18] L. Buttyán and J. Hubaux, *Security and cooperation in wireless networks.* Cambridge University Press, 2007.

[19] S. Capkun and J.-P. Hubaux, "Secure positioning in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 221–232, 2006.

[20] G. Carpenter, S. Grossberg, N. Markuzon, J. Reynolds, and D. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 698–713, 1992.

[21] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *IEE Proceedings Radar, Sonar and Navigation*, vol. 146, no. 1, pp. 2–7, 1999.

[22] P. Castro, P. Chiu, T. Kremenek, and R. Muntz, "A probabilistic room location service for wireless networked environments," in *ACM International Conference on Ubiquitous Computing (Ubicomp)*, 2001, pp. 18–34.

[23] X. Chai and Q. Yang, "Reducing the calibration effort for location estimation using unlabeled samples," in *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2005, pp. 95–104.

[24] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yazti, "Crowdsourcing with smartphones," *IEEE Internet Computing*, vol. 16, no. 5, pp. 36–44, 2012.

[25] C.-F. Chen, "Bayesian inference for a normal dispersion matrix and its application to stochastic multiple regression analysis," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 41, no. 2, pp. 235–248, 1979.

[26] H. Chen and K. Sezaki, "Distributed target tracking algorithm for wireless sensor networks," in *IEEE International Conference on Communications (ICC)*, 2011, pp. 1–5.

[27] J. Chen, R. Hudson, and K. Yao, "Maximum-likelihood source localization and unknown sensor location estimation for wideband signals in the near-field," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1843–1854, Aug. 2002.

[28] W. Chen, J. Hou, and L. Sha, "Dynamic clustering for acoustical target tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 3, pp. 258–271, Jul.-Sep. 2004.

[29] Y. Chen, W. Xu, W. Trappe, and Y. Zhang, "Attack detection in wireless localization," *Securing Emerging Wireless Systems*, pp. 1–22, 2009.

[30] Y. Chen, K. Kleisouris, X. Li, and R. P. Martin, "The robustness of localization algorithms to signal strength attacks: a comparative study," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2006, pp. 546–563.

[31] Y. Chen, W. Trappe, and R. Martin, "Attack detection in wireless localization," in *26th IEEE International Conference on Computer Communications INFOCOM*, 2007, pp. 1964–1972.

[32] Y. Chen, Q. Yang, J. Yin, and X. Chai, "Power-efficient access-point selection for indoor location estimation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 7, pp. 877–888, 2006.

[33] Y. Chen and H. Kobayashi, "Signal strength based indoor geolocation," in *IEEE International Conference on Communications (ICC)*, vol. 1, 2002, pp. 436–439.

[34] H. Cheng, H.-y. Luo, and F. Zhao, "Device-clustering algorithm in crowdsourcing-based localization," *The Journal of China Universities of Posts and Telecommunications*, vol. 19, Supplement 2, no. 0, pp. 114–121, Oct. 2012.

[35] Y.-C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm, "Accuracy characterization for metropolitan-scale Wi-Fi localization," in *3rd ACM International conference on Mobile systems, applications, and services (MobiSys)*, 2005, pp. 233–245.

[36] K. Chintalapudi, A. P. Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *16th international conference on Mobile computing and networking (MobiCom)*, 2010, pp. 173–184.

[37] A. Czarlinska, W. Luh, and D. Kundur, "Attacks on sensing in hostile wireless sensor-actuator environments," in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2007, pp. 1001–1005.

[38] M. Ding, F. Liu, A. Thaeler, D. Chen, and X. Cheng, "Fault-tolerant target localization in sensor networks," *EURASIP Journal on Wireless Communications and Networking*, 2007.

[39] P. M. Djuric, M. Vemula, and M. F. Bugallo, "Target tracking by particle filtering in binary sensor networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2229–2238, 2008.

[40] F. Dong, Y. Chen, J. Liu, Q. Ning, and S. Piao, "A calibration-free localization solution for handling signal strength variance," in *2nd international conference on Mobile Entity Localization and Tracking in GPS-less environments (MELT)*, 2009, pp. 79–90.

[41] R. Empson, "Apple acquires indoor GPS startup WiFiSlam for $20M," Techcrunch.com, March 2013. [Online]. Available: http://goo.gl/WI3g6

[42] F. Evennou, F. Marx, and E. Novakov, "Map-aided indoor mobile positioning system using particle filter," in *IEEE Wireless Communications and Networking Conference (WCNC)*, vol. 4, 2005, pp. 2490–2494.

[43] S.-H. Fang and T.-N. Lin, "Indoor location system based on discriminant-adaptive neural network in IEEE 802.11 environments," *IEEE Transactions on Neural Networks*, vol. 19, no. 11, pp. 1973–1978, 2008.

[44] S.-H. Fang, C.-H. Wang, S.-M. Chiou, and P. Lin, "Calibration-free approaches for robust Wi-Fi positioning against device diversity: A performance comparison," in *IEEE 75th Vehicular Technology Conference (VTC Spring)*, 2012, pp. 1–5.

[45] A. Farruggia, G. Lo Re, and M. Ortolani, "Detecting faulty wireless sensor nodes through stochastic classification," in *IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2011, pp. 148–153.

[46] C. Figuera, J. Rojo-Alvarez, I. Mora-Jimenez, A. Guerrero-Curieses, M. Wilby, and J. Ramos-Lopez, "Time-space sampling and mobile device calibration for WiFi indoor location systems," *IEEE Transactions on Mobile Computing*, vol. 10, no. 7, pp. 913–926, 2011.

[47] V. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello, "Bayesian filtering for location estimation," *IEEE Pervasive Computing*, vol. 2, no. 3, pp. 24–33, 2003.

[48] T. Garcia-Valverde, A. Garcia-Sola, A. Gomez-Skarmeta, J. Botia, H. Hagras, J. Dooley, and V. Callaghan, "An adaptive learning fuzzy logic system for indoor localisation using Wi-Fi in ambient intelligent environments," in *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2012, pp. 1–8.

[49] T. Garcia-Valverde, A. Garcia-Sola, H. Hagras, J. Dooley, V. Callaghan, and J. Botia, "A fuzzy logic-based system for indoor localization using WiFi in ambient intelligent environments," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 4, pp. 702–718, 2013.

[50] J. geun Park, B. Charrow, D. Curtis, J. Battat, E. Minkov, J. Hicks, S. Teller, and J. Ledlie, "Growing an organic indoor location system," in *8th ACM international conference on Mobile systems, applications, and services (MobiSys)*, 2010, pp. 271–284.

[51] A. Goswami, L. E. Ortiz, and S. R. Das, "WiGEM: a learning-based approach for indoor localization," in *7th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*. ACM, 2011, pp. 1–12.

[52] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki, "Practical robust localization over large-scale 802.11 wireless networks," in *10th international conference on Mobile computing and networking (MobiCom)*, 2004, pp. 70–84.

[53] R. U. Haider, C. Laoudias, and C. G. Panayiotou, "Health monitoring of WLAN localization infrastructure using smartphone inertial sensors," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2013.

[54] A. Hatami and K. Pahlavan, "A comparative performance evaluation of RSS-based positioning algorithms used in WLAN networks," in *IEEE Wireless Communications and Networking Conference (WCNC)*, vol. 4, 2005, pp. 2331–2337.

[55] V. Honkavirta, T. Perälä, S. Ali-Löytty, and R. Piché, "A comparative survey of WLAN location fingerprinting methods," in *6th Workshop on Positioning, Navigation and Communication (WPNC)*, 2009, pp. 243–251.

[56] A. Hossain and W.-S. Soh, "Cramer-Rao bound analysis of localization using signal strength difference as location fingerprint," in *IEEE International Conference on Computer Communications INFOCOM*, 2010, pp. 1–9.

[57] K. Kaemarungsi and P. Krishnamurthy, "Modeling of indoor positioning systems based on location fingerprinting," in *IEEE International Conference on Computer Communications INFOCOM*, vol. 2, 2004, pp. 1012–1022.

[58] N. Katenka, E. Levina, and G. Michailidis, "Local vote decision fusion for target detection in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 329–338, 2008.

[59] ——, "Robust target localization from binary decisions in wireless sensor networks," *Technometrics*, vol. 50, no. 4, pp. 448–461, 2008.

[60] W. Kim, K. Mechitov, J. Y. Choi, and S. Ham, "On target tracking with binary proximity sensors," in *4th International Symposium on Information Processing in Sensor Networks (IPSN)*, 2005, pp. 301–308.

[61] Y. Kim, H. Shin, and H. Cha, "Smartphone-based Wi-Fi pedestrian-tracking system tolerating the RSS variance problem," in *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2012, pp. 11–19.

[62] M. Kjærgaard, "Automatic mitigation of sensor variations for signal strength based location systems," in *2nd international conference on Location-and Context-Awareness (LoCA)*, 2006, pp. 30–47.

[63] ——, "A taxonomy for radio location fingerprinting," in *3rd international conference on Location-and Context-Awareness (LoCA)*, 2007, pp. 139–156.

[64] M. B. Kjærgaard, "Indoor location fingerprinting with heterogeneous clients," *Pervasive and Mobile Computing*, vol. 7, no. 1, pp. 31–43, 2011.

[65] A. Konstantinidis, G. Chatzimilioudis, C. Laoudias, S. Nicolaou, and D. Zeinalipour-Yazti, "Towards planet-scale localization on smartphones with a partial radiomap," in *4th ACM international workshop on Hot topics in planet-scale measurement*, 2012, pp. 9–14.

[66] L. Koski, T. Perälä, and R. Piché, "Indoor positioning using WLAN coverage area estimates," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010, pp. 1–7.

[67] A. Kotanen, M. Hannikainen, H. Leppakoski, and T. Hamalainen, "Positioning with IEEE 802.11b wireless LAN," in *14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications (PIMRC)*, vol. 3, 2003, pp. 2218–2222.

[68] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli, "On-line fault detection of sensor measurements," in *Proceedings of IEEE Sensors*, vol. 2, 2003, pp. 974–979.

[69] A. Kushki, K. Plataniotis, and A. Venetsanopoulos, "Sensor selection for mitigation of RSS-based attacks in wireless local area network positioning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 2065–2068.

[70] A. Kushki, K. N. Plataniotis, and A. N. Venetsanopoulos, "Kernel-based positioning in wireless local area networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 689–705, 2007.

[71] A. M. Ladd, K. E. Bekris, A. Rudys, L. E. Kavraki, and D. S. Wallach, "Robotics-based location sensing using wireless ethernet," *Wireless Networks*, vol. 11, no. 1, pp. 189–204, 2005.

[72] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit, "Place Lab: Device positioning using radio beacons in the wild," *Pervasive Computing*, pp. 116–133, 2005.

[73] C. Laoudias, P. Kemppi, and C. G. Panayiotou, "Localization using radial basis function networks and signal strength fingerprints in WLAN," in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2009, pp. 1–6.

[74] C. Laoudias, M. P. Michaelides, and C. G. Panayiotou, "Fault tolerant positioning using WLAN signal strength fingerprints," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010, pp. 1–8.

[75] C. Laoudias, M. Michaelides, and C. Panayiotou, "Fault tolerant fingerprint-based positioning," in *IEEE International Conference on Communications (ICC)*, 2011, pp. 1–5.

[76] ——, "ftTRACK: Fault tolerant target tracking in binary sensor networks," *ACM Transactions on Sensor Networks*, 2014.

[77] C. Laoudias, C. G. Panayiotou, and P. Kemppi, "On the RBF-based positioning using WLAN signal strength fingerprints," in *7th Workshop on Positioning Navigation and Communication (WPNC)*, 2010, pp. 93–98.

[78] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, and C. G. Panayiotou, "A platform for the evaluation of fingerprint positioning algorithms on android smartphones – demo," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2011.

[79] ——, "The Airplace indoor positioning platform for android smartphones," in *13th IEEE International Conference on Mobile Data Management (MDM)*, 2012, pp. 312–315.

[80] ——, "Demo: The Airplace indoor positioning platform," in *10th ACM international conference on Mobile systems, applications, and services (MobiSys)*, 2012, pp. 467–468.

[81] C. Laoudias, D. Eliades, P. Kemppi, C. Panayiotou, and M. Polycarpou, "Indoor localization using neural networks with location fingerprints," in *International Conference on Artificial Neural Networks (ICANN)*, 2009, pp. 954–963.

[82] C. Laoudias, G. Larkou, D. Zeinalipour-Yazti, and C. G. Panayiotou, "Airplace: Indoor geolocation on smartphones through WiFi fingerprinting," *ERCIM News*, vol. 2013, no. 93, 2013.

[83] C. Laoudias, M. P. Michaelides, and C. Panayiotou, "Fault tolerant target localization and tracking in binary WSNs using sensor health state estimation," in *IEEE International Conference on Communications (ICC)*, 2013, pp. 1469–1473.

[84] ——, "Sensor health state estimation for target tracking with binary sensor networks," in *IEEE International Conference on Communications (ICC)*, 2013, pp. 1878–1882.

[85] C. Laoudias, M. P. Michaelides, and C. G. Panayiotou, "Fault detection and mitigation in WLAN RSS fingerprint-based positioning," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2011, pp. 1–7.

[86] ——, "Fault detection and mitigation in WLAN RSS fingerprint-based positioning," *Journal of Location Based Services*, vol. 6, no. 2, pp. 101–116, Jun. 2012.

[87] C. Laoudias, R. Piché, and C. G. Panayiotou, "Device signal strength self-calibration using histograms," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2012, pp. 1–8.

[88] ——, "Device self-calibration in location systems using signal strength histograms," *Journal of Location Based Services*, vol. 7, no. 3, pp. 165–181, Aug. 2013.

[89] C. Laoudias, D. Zeinalipour-Yazti, and C. G. Panayiotou, "Crowdsourced indoor localization for diverse devices through radiomap fusion," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2013.

[90] J. Ledlie, J.-g. Park, D. Curtis, A. Cavalcante, L. Camara, A. Costa, and R. Vieira, "Molé: a scalable, user-generated WiFi positioning engine," *Journal of Location Based Services*, vol. 6, no. 2, pp. 55–80, Jun. 2012.

[91] H. Lee, M. Wicke, B. Kusy, and L. Guibas, "Localization of mobile users using trajectory matching," in *1st ACM international workshop on Mobile entity localization and tracking in GPS-less environments (MELT)*, 2008, pp. 123–128.

[92] M. Lee, S. H. Jung, S. Lee, and D. Han, "Elekspot: A platform for urban place recognition via crowdsourcing," in *IEEE/IPSJ 12th International Symposium on Applications and the Internet (SAINT)*, 2012, pp. 190–195.

[93] M. Lee, H. Yang, D. Han, and C. Yu, "Crowdsourced radiomap for room-level place recognition in urban environment," in *8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2010, pp. 648–653.

[94] B. Li, J. Salter, A. Dempster, and C. Rizos, "Indoor positioning techniques based on wireless LAN," in *1st IEEE International Conference on Wireless Broadband and Ultra Wideband Communications*, 2006, pp. 13–16.

[95] Z. Li, W. Trappe, Y. Zhang, and B. Nath, "Robust statistical methods for securing wireless localization in sensor networks," in *International Symposium on Information Processing in Sensor Networks (IPSN)*, 2005, pp. 91–98.

[96] H. Lim, L.-C. Kung, J. C. Hou, and H. Luo, "Zero-configuration, robust indoor localization: Theory and experimentation," in *25th IEEE International Conference on Computer Communications INFOCOM*, 2006, pp. 1–12.

[97] K. Lorincz and M. Welsh, "MoteTrack: a robust, decentralized approach to RF-based location tracking," *Personal and Ubiquitous Computing*, vol. 11, no. 6, pp. 489–503, Aug. 2007.

[98] X. Luo, M. Dong, and Y. Huang, "On distributed fault-tolerant detection in wireless sensor networks," *IEEE Transactions on Computers*, vol. 55, no. 1, pp. 58–70, 2006.

[99] J. Machaj, P. Brida, and R. Piché, “Rank based fingerprinting algorithm for indoor positioning,” in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2011, pp. 1–6.

[100] D. Madigan, E. Einahrawy, R. Martin, W.-H. Ju, P. Krishnan, and A. Krishnakumar, “Bayesian indoor positioning systems,” in *24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 2, 2005, pp. 1217–1227.

[101] A. Mahtab Hossain, Y. Jin, W.-S. 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.

[102] A. Mahtab Hossain, H. N. Van, Y. Jin, and W.-S. Soh, “Indoor localization using multiple wireless technologies,” in *IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, 2007, pp. 1–8.

[103] M. P. Michaelides and C. G. Panayiotou, “Fault tolerant maximum likelihood event localization in sensor networks using binary data,” *IEEE Signal Processing Letters*, vol. 16, no. 5, pp. 406–409, 2009.

[104] M. P. Michaelides, C. Laoudias, and C. G. Panayiotou, “Fault tolerant detection and tracking of multiple sources in WSNs using binary data,” in *48th IEEE Conference on Decision and Control (CDC)*, 2009, pp. 3769–3774.

[105] M. P. Michaelides and C. G. Panayiotou, “Event detection using sensor networks,” in *45th IEEE Conference on Decision and Control (CDC)*, 2006, pp. 6784–6789.

[106] M. Michaelides, C. Laoudias, and C. Panayiotou, “Fault tolerant target localization and tracking in wireless sensor networks using binary data,” in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2011, pp. 1–6.

[107] ——, “Fault tolerant localization and tracking of multiple sources in WSNs using binary data,” *IEEE Transactions on Mobile Computing*, 2014.

[108] M. Michaelides and C. Panayiotou, “SNAP: Fault tolerant event location estimation in sensor networks using binary data,” *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1185–1197, 2009.

[109] P. Misikangas and L. Lekman, “Applications of signal quality observations,” WO Patent WO/2004/008,796, 2005.

[110] C. Nerguizian, C. Despins, and S. Affes, “Indoor geolocation with received signal strength fingerprinting technique and neural networks,” *Telecommunications and Networking - ICT 2004*, pp. 866–875, 2004.

[111] K. Ni and G. Pottie, “Bayesian selection of non-faulty sensors,” in *IEEE International Symposium on Information Theory (ISIT)*, 2007, pp. 616–620.

[112] K. Ni, N. Ramanathan, M. N. H. Chehade, L. Balzano, S. Nair, S. Zahedi, E. Kohler, G. Pottie, M. Hansen, and M. Srivastava, “Sensor network data fault types,” *ACM Transactions on Sensor Networks*, vol. 5, no. 3, pp. 1–29, 2009.

[113] R. Niu and P. Varshney, “Target location estimation in wireless sensor networks using binary data,” in *38th Annual Conference on Information Sciences and Systems*, 2004.

[114] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *46th IEEE Conference on Decision and Control (CDC)*, 2007, pp. 5492–5498.

[115] J. J. Pan, J. Kwok, Q. Yang, and Y. Chen, "Multidimensional vector regression for accurate and low-cost location estimation in pervasive computing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 9, pp. 1181–1193, 2006.

[116] J. Park, D. Curtis, S. Teller, J. Ledlie *et al.*, "Implications of device diversity for organic localization," in *IEEE International Conference on Computer Communications INFOCOM*, 2011, pp. 3182–3190.

[117] L. Petrou, G. Larkou, C. Laoudias, D. Zeinalipour-Yazti, and C. G. Panayiotou, "Anyplace: Indoor positioning and navigation in the big-data era – demo," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2013.

[118] ——, "Demonstration abstract: Crowdsourced indoor localization and navigation with anyplace," in *13th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2014.

[119] R. Piché, "Stochastic processes," 2010. [Online]. Available: http://urn.fi/URN:NBN:fi:tty-201012021377

[120] P. Prasithsangaree, P. Krishnamurthy, and P. Chrysanthis, "On indoor position location with wireless LANs," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, vol. 2, 2002, pp. 720–724.

[121] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in *18th ACM international conference on Mobile computing and networking (MobiCom)*, 2012, pp. 293–304.

[122] T. Rappaport, *Wireless communications.* Prentice Hall PTR New Jersey, 2002.

[123] M. Raspopoulos, C. Laoudias, L. Kanaris, A. Kokkinis, C. Panayiotou, and S. Stavrou, "3D Ray Tracing for device-independent fingerprint-based positioning in WLANs," in *9th Workshop on Positioning Navigation and Communication (WPNC)*, 2012, pp. 109–113.

[124] M. Raspopoulos, C. Laoudias, L. Kanaris, A. Kokkinis, C. G. Panayiotou, and S. Stavrou, "Cross device fingerprint-based positioning using 3D Ray Tracing," in *8th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2012, pp. 147–152.

[125] T. Roos, P. Myllymaki, H. Tirri, P. Misikangas, and J. Sievanen, "A probabilistic approach to WLAN user location estimation," *International Journal of Wireless Information Networks*, vol. 9, no. 3, pp. 155–164, Jul. 2002.

[126] S. Saha, K. Chaudhuri, D. Sanghi, and P. Bhagwat, "Location determination of a mobile device using IEEE 802.11b access point signals," in *IEEE Wireless Communications and Networking Conference (WCNC)*, vol. 3, 2003, pp. 1987–1992.

[127] X. Sheng and Y.-H. Hu, "Energy based acoustic source localization," in *2nd International Conference on Information Processing in Sensor Networks (IPSN)*, April 2003, pp. 286–300.

[128] N. Shrivastava, R. Mudumbai, U. Madhow, and S. Suri, "Target tracking with binary proximity sensors," *ACM Transactions on Sensor Networks*, vol. 5, no. 4, pp. 1–33, 2009.

[129] J. Singh, R. Kumar, U. Madhow, S. Suri, and R. Cagley, "Multiple-target tracking with binary proximity sensors," *ACM Transactions on Sensor Networks*, vol. 8, no. 1, pp. 1–26, 2011.

[130] R. Singh, L. Macchi, C. Regazzoni, and K. Plataniotis, "A statistical modelling based location determination method using fusion in WLAN," in *International Workshop Wireless Ad-Hoc Networks*, 2005.

[131] Z. Sun, Y. Chen, J. Qi, and J. Liu, "Adaptive localization through transfer learning in indoor Wi-Fi environment," in *7th International Conference on Machine Learning and Applications (ICMLA)*, 2008, pp. 331–336.

[132] A. S. Tanenbaum, C. Gamage, and B. Crispo, "Taking sensor networks from the lab to the jungle," *IEEE Computer*, vol. 39, no. 8, pp. 98–100, 2006.

[133] J. Teng, H. Snoussi, and C. Richard, "Decentralized variational filtering for target tracking in binary sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 10, pp. 1465–1477, 2010.

[134] A. Teuber, B. Eissfeller, and T. Pany, "A two-stage fuzzy logic approach for wireless LAN indoor positioning," in *IEEE/ION Position, Location, And Navigation Symposium (PLANS)*, 2006, pp. 730–738.

[135] A. Teuber and B. Eissfeller, "WLAN indoor positioning based on euclidean distances and fuzzy logic," in *3rd Workshop on Positioning, Navigation and Communication (WPNC)*, 2006, pp. 159–168.

[136] G. Theodorakopoulos and J. S. Baras, "On trust models and trust evaluation metrics for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 318–328, 2006.

[137] N. O. Tippenhauer, K. B. Rasmussen, C. Pöpper, and S. Čapkun, "Attacks on public WLAN-based positioning systems," in *7th ACM International Conference on Mobile systems, applications, and services (MobiSys)*, 2009, pp. 29–40.

[138] 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.

[139] M. Vuran, O. Akan, and I. Akyildiz, "Spatio-temporal correlation: theory and applications for wireless sensor networks," *Computer Networks*, vol. 45, pp. 245–259, March 2004.

[140] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *6th ACM International Symposium on Mobile ad hoc networking and computing*, 2005, pp. 46–57.

[141] X. Xu, X. Gao, J. Wan, and N. Xiong, "Trust index based fault tolerant multiple event localization algorithm for WSNs," *Sensors*, vol. 11, no. 7, pp. 6555–6574, 2011.

[142] S. Yang, P. Dessai, M. Verma, and M. Gerla, "FreeLoc: Calibration-free crowdsourced indoor localization," in *IEEE International Conference on Computer Communications INFOCOM*, 2013, pp. 2481–2489.

[143] H.-C. Yen and C.-C. Wang, "Adapting gaussian processes for cross-device Wi-Fi localization," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2013.

[144] W. Yeung, J. Zhou, and J. Ng, "Enhanced fingerprint-based location estimation system in wireless LAN environment," *Emerging Directions in Embedded and Ubiquitous Computing*, pp. 273–284, 2007.

[145] J. Yim, "Introducing a decision tree-based indoor positioning technique," *Expert Systems with Applications*, vol. 34, no. 2, pp. 1296–1302, Feb. 2008.

[146] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *3rd ACM International Conference on Mobile systems, applications, and services (MobiSys)*, 2005, pp. 205–218.

[147] D. Zeinalipour-Yazti, C. Laoudias, C. Costa, M. Vlachos, M. I. Andreou, and D. Gunopulos, "Crowdsourced trace similarity with smartphones," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 6, pp. 1240–1253, 2013.

[148] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann, 2004.

[149] V. W. Zheng, S. J. Pan, Q. Yang, and J. J. Pan, "Transferring multi-device localization models using latent multi-task learning," in *23rd national conference on Artificial intelligence*, vol. 3. AAAI Press, 2008, pp. 1427–1432.

Appendix A

Maximum Likelihood Sensor State Estimate

The maximum likelihood sensor state estimate in case the sensor output is *wrong*, i.e. $r_n(t) = 1$, is given by

$$\hat{s}_n(t+1)|_{r_n(t)=1} = \begin{cases} H & \text{if } \hat{\pi}_n^H(t) > \frac{p_n^f(t)}{p_n^f(t)+p_n^h(t)} \\ F & \text{otherwise.} \end{cases}$$

Proof. From equation (3.8), estimating the sensor state as *Healthy* in case the sensor output is *wrong* implies that

$$\begin{aligned} \hat{\pi}_n^{H|1}(t) > \hat{\pi}_n^{F|1}(t) &\Leftrightarrow \frac{p_n^h(t)\cdot\hat{\pi}_n^H(t)}{p_n^f(t)\cdot\hat{\pi}_n^H(t)+p_n^h(t)\cdot\hat{\pi}_n^H(t)} > \frac{p_n^f(t)\cdot\hat{\pi}_n^F(t)}{p_n^f(t)\cdot\hat{\pi}_n^F(t)+p_n^h(t)\cdot\hat{\pi}_n^H(t)} \\ &\Leftrightarrow p_n^h(t)\cdot\hat{\pi}_n^H(t) > p_n^f(t)\cdot\hat{\pi}_n^F(t) \\ &\Leftrightarrow p_n^h(t)\cdot\hat{\pi}_n^H(t) > p_n^f(t)\cdot(1-\hat{\pi}_n^H(t)) \\ &\Leftrightarrow \hat{\pi}_n^H(t) > \frac{p_n^f(t)}{p_n^f(t)+p_n^h(t)}. \end{aligned}$$

Otherwise, since the sensor output is *wrong*, its state is estimated as *Faulty*. □

The maximum likelihood sensor state estimate in case the sensor output is *correct*, i.e $r_n(t) = 0$, is given by

$$\hat{s}_n(t+1)|_{r_n(t)=0} = \begin{cases} F & \text{if } \hat{\pi}_n^H(t) < \frac{1-p_n^f(t)}{2-p_n^f(t)-p_n^h(t)} \\ H & \text{otherwise.} \end{cases}$$

Proof. From equation (3.8), estimating the sensor state as *Faulty* in case the sensor output is *correct* implies that

$$\begin{aligned}
\hat{\pi}_n^{H|0}(t) < \hat{\pi}_n^{F|0}(t) \Leftrightarrow & \frac{(1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)}{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t) + (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)} < \\
& < \frac{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t)}{(1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t) + (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t)} \\
& \Leftrightarrow (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t) < (1 - p_n^f(t)) \cdot \hat{\pi}_n^F(t) \\
& \Leftrightarrow (1 - p_n^h(t)) \cdot \hat{\pi}_n^H(t) < (1 - p_n^f(t)) \cdot (1 - \hat{\pi}_n^H(t)) \\
& \Leftrightarrow \hat{\pi}_n^H(t) - p_n^h(t) \cdot \hat{\pi}_n^H(t) < 1 - \hat{\pi}_n^H(t) - p_n^f(t) + p_n^f(t) \cdot \hat{\pi}_n^H(t) \\
& \Leftrightarrow \hat{\pi}_n^H(t) < \frac{1 - p_n^f(t)}{2 - p_n^f(t) - p_n^h(t)}.
\end{aligned}$$

Otherwise, since the sensor output is *correct*, its state is estimated as *Healthy*. □

Appendix B

Sensor Health State Estimation Algorithms

ALGORITHM 1: Static Sensor Health State Estimator

```
Input: Target location $\tilde{\ell}_s(t)$, sensor locations $\ell_n$, alarm status $A_n(t)$, ROI radius $R_I$.
Output: Estimated sensor state $\hat{s}_n(t+1)$.

$\hat{s}_n(0) = H$;
for each sensor node n do
    Determine the error signal $\tilde{r}_n(t)$;
    Calculate $\hat{C}_n^T(t)$ using equation (3.16) in equation (3.6);
    Calculate $[\hat{\pi}_n^F \ \hat{\pi}_n^H]^T$ with equation (3.15);
    if $\tilde{r}_n(t) = 1$ then
        if $\hat{\pi}_n^H > \frac{p_n^f(t)}{p_n^f(t)+p_n^h(t)}$ then
            $\hat{s}_n(t+1) = H$;
        else
            $\hat{s}_n(t+1) = F$;
        end
    else
        if $\hat{\pi}_n^H < \frac{1-p_n^f(t)}{2-p_n^f(t)-p_n^h(t)}$ then
            $\hat{s}_n(t+1) = F$;
        else
            $\hat{s}_n(t+1) = H$;
        end
    end
end
```

ALGORITHM 2: Dynamic Sensor Health State Estimator

```
Input: Target location $\tilde{\ell}_s(t)$, sensor locations $\ell_n$, alarm status $A_n(t)$, ROI radius $R_I$.
Output: Estimated sensor state $\hat{s}_n(t+1)$.
1  $\hat{s}_n(0) = H$; $[\hat{\pi}_n^F(0)\ \hat{\pi}_n^H(0)]^T = [0.5\ 0.5]$;
2  for each sensor node n do
3      Determine the error signal $\tilde{r}_n(t)$;
4      if $\tilde{r}_n(t) = 1$ then
5          if $\hat{\pi}_n^H(t) > \frac{p_n^f(t)}{p_n^f(t)+p_n^h(t)}$ then
6              $\hat{s}_n(t+1) = H$;
7          else
8              $\hat{s}_n(t+1) = F$;
9          end
10         Calculate $\hat{\pi}_n^{F|1}(t)$ with equation (3.10a) and $\hat{\pi}_n^{H|1}(t)$ with equation (3.10b);
11         Predict $[\hat{\pi}_n^F(t+1)\ \hat{\pi}_n^H(t+1)]^T = \hat{C}_n^T(t)[\hat{\pi}_n^{F|1}(t)\ \hat{\pi}_n^{H|1}(t)]^T$;
12     else
13         if $\hat{\pi}_n^H(t) < \frac{1-p_n^f(t)}{2-p_n^f(t)-p_n^h(t)}$ then
14             $\hat{s}_n(t+1) = F$;
15         else
16             $\hat{s}_n(t+1) = H$;
17         end
18         Calculate $\hat{\pi}_n^{F|0}(t)$ with equation (3.10c) and $\hat{\pi}_n^{H|0}(t)$ with equation (3.10d);
19         Predict $[\hat{\pi}_n^F(t+1)\ \hat{\pi}_n^H(t+1)]^T = \hat{C}_n^T(t)[\hat{\pi}_n^{F|0}(t)\ \hat{\pi}_n^{H|0}(t)]^T$;
20     end
21 end
```

ALGORITHM 3: ftTRACK Target Tracking

```
Input: Sensor locations $\ell_n$, alarm status $A_n(t)$, ROI radius $R_I$.
Output: Estimated target location $\tilde{\ell}_s(t)$.
1  $\hat{s}_n(0) = H$; $\{X^i(0)\}_{i=1}^{N_p} = [\hat{\ell}_s(0)\ 0\ 0]^T$; $\{\omega^i(0)\}_{i=1}^{N_p} = \frac{1}{N_p}$;
2  while $t > 0$ do
       // Prediction step
3      $X^i(t) = \Phi X^i(t-1) + \Gamma W(t-1)$;
       // Target localization
4      $\left(\hat{\ell}_s(t), \hat{e}_s(t)\right) = Localization(\hat{s}_n(t), A_n(t))$;
       // Update step
5      $\omega^i(t) = \omega^i(t-1)p(\hat{\ell}_s(t)|X^i(t))$;
6      $\omega^i(t) = \frac{\omega^i(t)}{\sum_{i=1}^{N_p}\omega^i(t)}$;
7      $\{\omega^i(t)\}_{i=1}^{N_p} = LinearTimeResampling(\{\omega^i(t)\}_{i=1}^{N_p})$;
8      $\tilde{\ell}_s(t) = \sum_{i=1}^{N_p}\omega^i(t)X^i(t)$;
       // Sensor Health State Estimation
9      $\hat{s}_n(t+1) = SensorHealthStateEstimator(\ell_n, \tilde{\ell}_s(t), A_n(t), R_I)$
10 end
```

Appendix C

Calculation of the Probabilities $p_n^h(t)$ and $p_n^f(t)$

During target tracking, $d_n(t)$ and $\tilde{d}_n(t)$ denote the distance of sensor n from the actual target location $\ell_s(t)$ and the estimated target location $\tilde{\ell}_s(t)$, respectively. Due to the noise $w_n(t) \sim \mathcal{N}(0, \sigma_w^2)$ disturbing the sensor measurements, as well as the presence of faults that degrade the localization accuracy, $d_n(t)$ and $\tilde{d}_n(t)$ are not necessarily equal. We assume that $d_n(t)$ is a normally distributed random variable $d_n(t) \sim \mathcal{N}(\tilde{d}_n(t), \sigma_d^2)$, where σ_d reflects the uncertainty in the estimated target location. In this context, the probability $p_n^h(t)$ of sensor n having a *wrong* output at time t given that its state is *Healthy* can be calculated as:

$$\begin{aligned} p_n^h(t) &= \mathbf{P}[r_n(t) = 1 | s_n(t) = H] \\ &= \mathbf{P}[A_n(t) = 0 | d_n(t) \leq R_I]\mathbf{P}[d_n(t) \leq R_I] + \mathbf{P}[A_n(t) = 1 | d_n(t) > R_I]\mathbf{P}[d_n(t) > R_I] \\ &= \mathbf{P}[z_n(t) \leq T | d_n(t) \leq R_I]\mathbf{P}[d_n(t) \leq R_I] + \mathbf{P}[z_n(t) > T | d_n(t) > R_I]\mathbf{P}[d_n(t) > R_I] \\ &= \left[1 - Q\left(\frac{T - \mu_n(t)}{\sigma_w}\right)\right]\left[1 - Q\left(\frac{R_I - \tilde{d}_n(t)}{\sigma_d}\right)\right] + Q\left(\frac{T - \mu_n(t)}{\sigma_w}\right)Q\left(\frac{R_I - \tilde{d}_n(t)}{\sigma_d}\right), \end{aligned}$$

where $\mu_n(t) = \frac{c}{1+\tilde{d}_n(t)^\gamma}$ and $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty \exp(-\frac{y^2}{2})dy$ is the right-tail probability of a Gaussian random variable $\mathcal{N}(0, 1)$.

In a similar way, the probability $p_n^f(t)$ of sensor n having a *wrong* output at time t given that its state is *Faulty* can be calculated as:

$$\begin{aligned} p_n^f(t) &= \mathbf{P}[r_n(t) = 1 | s_n(t) = F] \\ &= \mathbf{P}[A_n(t) = 0 | d_n(t) > R_I]\mathbf{P}[d_n(t) > R_I] + \mathbf{P}[A_n(t) = 1 | d_n(t) \leq R_I]\mathbf{P}[d_n(t) \leq R_I] \\ &= \mathbf{P}[z_n(t) \leq T | d_n(t) > R_I]\mathbf{P}[d_n(t) > R_I] + \mathbf{P}[z_n(t) > T | d_n(t) \leq R_I]\mathbf{P}[d_n(t) \leq R_I] \\ &= \left[1 - Q\left(\frac{T - \mu_n(t)}{\sigma_w}\right)\right]Q\left(\frac{R_I - \tilde{d}_n(t)}{\sigma_d}\right) + Q\left(\frac{T - \mu_n(t)}{\sigma_w}\right)\left[1 - Q\left(\frac{R_I - \tilde{d}_n(t)}{\sigma_d}\right)\right]. \end{aligned}$$

Appendix D

Algorithms for Distributed ftTRACK

ALGORITHM 4: Distributed Fault Tolerant Leader Election Protocol (D-FTLEP)

```
   Input: Set of neighbouring alarmed sensor nodes.
   Output: Elected leader status.
1  All alarmed sensors broadcast an ALARM message;
2  Node $n$ calculates the function $F_n$ using the received ALARM messages from its neighbours;
3  if $F_n > 0$ then
4      Continue with next step;
5  else
6      STOP;
7  end
8  Wait for a period $h(1/F_n)$;
9  if during the waiting period a LEADER message with value $f \geq F_n$ is received then
10     STOP;
11 end
12 Broadcast LEADER message with value $F_n$ and assume leadership role;
```

ALGORITHM 5: Scoring Matrix Construction

```
   Input: $[X_n, Y_n, b_n]$ for sensor nodes $n = 1, \dots, N_l \in ROS_l$.
   Output: Scoring matrix $L_l$.
1  $L_l \leftarrow 0$;
2  for all cells $\mathcal{M}_l^{-1}(i,j) \in \mathcal{G}_l$ do
3      for all sensor nodes n that have cell $\mathcal{M}_l^{-1}(i,j) \in \overline{ROC_n}$ do
4          $L_l(i,j) \leftarrow L_l(i,j) + b_n$;
5      end
6  end
```

Appendix E

Fault Tolerant Fingerprinting Algorithms

ALGORITHM 6: Hybrid KNN (H-KNN) Localization

Input: Observed fingerprint $\mathbf{s}$, Fingerprints $\mathbf{r}_i$ in the radiomap, Fault detection threshold γ.
Output: Estimated user location $\widehat{\ell}(\mathbf{s})$.

1. **RSS Distance Calculation:** Use (4.6) to calculate the distances d_i between fingerprint $\mathbf{s}$ and all fingerprints in the radiomap $\mathbf{r}_i$.
2. **Fault Indicator Computation:** Compute the fault indicator $D_{sum}^{(2)}$ using d_i.
3. **Location Estimation:** If $D_{sum}^{(2)} > \gamma$ (i.e., a fault is signified), then calculate the distances d_i' with (4.7) and estimate location $\widehat{\ell}(\mathbf{s})$; otherwise use the distances d_i calculated in step (1) to determine location.

ALGORITHM 7: Hybrid MMSE (H-MMSE) Localization

Input: Observed fingerprint $\mathbf{s}$, Fingerprints $\mathbf{r}_i$ in the radiomap, Fault detection threshold γ.
Output: Estimated user location $\widehat{\ell}(\mathbf{s})$.

1. **Likelihood Calculation:** Use (4.8) to calculate the likelihood $p(\mathbf{s}|\ell_i)$ of $\mathbf{s}$ at each candidate location ℓ_i.
2. **Fault Indicator Computation:** Compute the fault indicator P_{max} with (4.4) using $p(\mathbf{s}|\ell_i)$.
3. **Location Estimation:** If the condition $P_{max} > \gamma$ is satisfied then calculate the likelihood $p'(\mathbf{s}|\ell_i)$ with (4.9) and estimate location $\widehat{\ell}(\mathbf{s})$; otherwise use $p(\mathbf{s}|\ell_i)$ calculated in step (1) to determine location.

ALGORITHM 8: Hybrid MED (H-MED) Localization

Input: Observed fingerprint **s**, Fingerprints $\mathbf{r}_i$ in the radiomap, Fault detection threshold γ.
Output: Estimated user location $\widehat{\ell}(\mathbf{s})$.

1. **RSS Distance Calculation:** Use (4.6) to calculate the distances d_i between fingerprint **s** and all fingerprints in the radiomap $\mathbf{r}_i$.
2. **Fault Indicator Computation:** Compute the fault indicator $D_{sum}^{(2)}$ using d_i.
3. **Location Estimation:** If $D_{sum}^{(2)} > \gamma$, then calculate the RSS distances with the median-based metric in (4.5) and calculate $\widehat{\ell}(\mathbf{s})$; otherwise use the distances d_i calculated in step (1) to determine location.

Appendix F

Least Squares Mapping with Inverse CDF Percentiles

If $\mathbf{u}$ is a continuous random variable and $\mathbf{y} = f(\mathbf{u})$ with monotonically increasing f then $f = F_{\mathbf{y}}^{-1} \circ F_{\mathbf{u}}$. In particular, the inverse CDF ordered pairs

$$\{(u_i, y_i) = (F_{\mathbf{u}}^{-1}(q_i), F_{\mathbf{y}}^{-1}(q_i)) : q_i \in \{0.1, \ldots, 0.9\}\}$$

lie on the curve $y = f(u)$.

Proof. We have

$$\begin{aligned} F_{\mathbf{u}}(u) &= P(\mathbf{u} \leq u) = P(f(\mathbf{u}) \leq f(u)) = \\ &= P(\mathbf{y} \leq f(u)) = F_{\mathbf{y}}(f(u)). \end{aligned}$$

Applying $F_{\mathbf{y}}^{-1}$ to both sides gives the identity $f = F_{\mathbf{y}}^{-1} \circ F_{\mathbf{u}}$. Also, the components of the inverse CDF ordered pairs satisfy

$$y_i = F_{\mathbf{y}}^{-1}(q_i) = F_{\mathbf{y}}^{-1}(F_{\mathbf{u}}(u_i)) = f(u_i).$$

□

Appendix G

Useful Lemmas for Differential Fingerprinting

We show that the following statement holds for DIFF differential fingerprints

$$\begin{aligned}\tilde{d}_i^2 &= \sum_{k=2}^{n}\sum_{j=1}^{k-1}\left(\tilde{r}_{ijk} - \tilde{s}_{jk}\right)^2\\ &= \sum_{k=2}^{n}\sum_{j=1}^{k-1}\left(r_{ij} - r_{ik} - (s_j - s_k)\right)^2\\ &= \frac{1}{2}\sum_{k=1}^{n}\sum_{j=1}^{n} B_{ijk}^2,\end{aligned}$$

where $B_{ijk} = r_{ij} - r_{ik} - (s_j - s_k)$.

Proof. From the definition of B_{ijk} we have

$$\begin{aligned} B_{ijk}^2 &= B_{ikj}^2 && \text{(G.1)}\\ B_{ijj}^2 &= B_{ikk}^2 = 0. && \text{(G.2)}\end{aligned}$$

Basis: We show that the statement holds for $n = 2$. Using equations G.1 and G.2, we can easily see that

$$\begin{aligned}\sum_{k=2}^{2}\sum_{j=1}^{k-1} B_{ijk}^2 = \frac{1}{2}\sum_{k=1}^{2}\sum_{j=1}^{2} B_{ijk}^2 &\Leftrightarrow\\ B_{i12}^2 = \frac{1}{2}\left(B_{i11}^2 + B_{i21}^2 + B_{i12}^2 + B_{i22}^2\right) &\Leftrightarrow\\ B_{i12}^2 = B_{i12}^2,\end{aligned}$$

thereby indeed the statement holds for $n = 2$.

Inductive step: We assume that the statement holds for $n = \lambda$, that is

$$\sum_{k=2}^{\lambda}\sum_{j=1}^{k-1} B_{ijk}^2 = \frac{1}{2}\sum_{k=1}^{\lambda}\sum_{j=1}^{\lambda} B_{ijk}^2. \tag{G.3}$$

It must then be shown that the statement also holds for $n = \lambda + 1$, that is

$$\sum_{k=2}^{\lambda+1}\sum_{j=1}^{k-1} B_{ijk}^2 = \frac{1}{2}\sum_{k=1}^{\lambda+1}\sum_{j=1}^{\lambda+1} B_{ijk}^2.$$

Splitting the outer sum and then using the induction hypothesis in equation G.3, the left-hand side can be rewritten to

$$\begin{aligned}
\sum_{k=2}^{\lambda+1}\sum_{j=1}^{k-1} B_{ijk}^2 &= \sum_{k=2}^{\lambda}\sum_{j=1}^{k-1} B_{ijk}^2 + \sum_{j=1}^{\lambda} B_{ij(\lambda+1)}^2 \\
&= \frac{1}{2}\sum_{k=1}^{\lambda}\sum_{j=1}^{\lambda} B_{ijk}^2 + \frac{1}{2}\sum_{j=1}^{\lambda} B_{ij(\lambda+1)}^2 + \frac{1}{2}\sum_{j=1}^{\lambda} B_{ij(\lambda+1)}^2 \\
&= \frac{1}{2}\sum_{k=1}^{\lambda+1}\sum_{j=1}^{\lambda} B_{ijk}^2 + \frac{1}{2}\sum_{j=1}^{\lambda} B_{ij(\lambda+1)}^2 \\
&= \frac{1}{2}\sum_{k=1}^{\lambda+1}\sum_{j=1}^{\lambda} B_{ijk}^2 + \frac{1}{2}\sum_{k=1}^{\lambda+1} B_{i(\lambda+1)k}^2 \\
&= \frac{1}{2}\sum_{k=1}^{\lambda+1}\sum_{j=1}^{\lambda+1} B_{ijk}^2,
\end{aligned}$$

thereby showing that indeed the statement holds for $n = \lambda + 1$, while for the last equality we have used equations G.1 and G.2.

Since both the basis and the inductive step have been performed, by mathematical induction, the statement holds for all natural n. □

We also show that the following holds

$$\sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2,$$

where $\mathbf{x} = [x_1, \dots, x_n]^T$ is a vector with mean value $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

Proof.

$$\begin{aligned}
\sum_{i=1}^{n}(x_i - \bar{x})^2 &= \sum_{i=1}^{n}(x_i^2 - 2x_i\bar{x} + \bar{x}^2) \\
&= \sum_{i=1}^{n} x_i^2 - 2\bar{x}\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} \bar{x}^2 \\
&= \sum_{i=1}^{n} x_i^2 - 2n\bar{x}^2 + n\bar{x}^2 \\
&= \sum_{i=1}^{n} x_i^2 - n\bar{x}^2.
\end{aligned}$$

□

Appendix H

Analytical Models for Differential Fingerprinting

H.1 Derivation of the MDF Analytical Model

In the case of MDF differential fingerprints, the NN localization method will return the correct location ℓ_1 only if the following condition is satisfied

$$\bar{\bar{d}}_1^2 \le \bar{\bar{d}}_2^2.$$

Using Theorem 3 the above condition is equivalent to

$$\begin{aligned}
\bar{\bar{d}}_1^2 \le \bar{\bar{d}}_2^2 &\Leftrightarrow d_1^2 - n(\bar{r}_1 - \bar{s})^2 \le d_d^2 - n(\bar{r}_2 - \bar{s})^2 \\
&\Leftrightarrow \sum_{j=1}^{n}(r_{1j} - s_j)^2 - n\bar{r}_1^2 + 2n\bar{r}_1\bar{s} - n\bar{s}^2 \le \sum_{j=1}^{n}(r_{2j} - s_j)^2 - n\bar{r}_2^2 + 2n\bar{r}_2\bar{s} - n\bar{s}^2 \\
&\Leftrightarrow \sum_{j=1}^{n} r_{1j}^2 - 2\sum_{j=1}^{n} r_{1j}s_j + \sum_{j=1}^{n} s_j^2 - n\bar{r}_1^2 + 2n\bar{r}_1\bar{s} \le \sum_{j=1}^{n} r_{2j}^2 - 2\sum_{j=1}^{n} r_{2j}s_j + \sum_{j=1}^{n} s_j^2 - n\bar{r}_2^2 + 2n\bar{r}_2\bar{s} \\
&\Leftrightarrow 2\sum_{j=1}^{n}\beta_j s_j + \sum_{j=1}^{n}(r_{1j}^2 - r_{2j}^2) - 2n(\bar{r}_2 - \bar{r}_1)\bar{s} + n(\bar{r}_2^2 - \bar{r}_1^2) \le 0 \\
&\Leftrightarrow 2\sum_{j=1}^{n}\beta_j s_j - 2n(\frac{1}{n}\sum_{j=1}^{n} r_{2j} - \frac{1}{n}\sum_{j=1}^{n} r_{1j})\bar{s} + \sum_{j=1}^{n}\delta_j \le 0 \\
&\Leftrightarrow 2\sum_{j=1}^{n}\beta_j s_j - 2\sum_{j=1}^{n}\beta_j\bar{s} + \sum_{j=1}^{n}\delta_j \le 0 \\
&\Leftrightarrow 2\sum_{j=1}^{n}\beta_j(s_j - \bar{s}) + \sum_{j=1}^{n}\delta_j \le 0,
\end{aligned}$$

where $\beta_j = (r_{2j} - r_{1j})$ and $\delta_j = (r_{1j}^2 - \bar{r}_1^2 - r_{2j}^2 + \bar{r}_2^2)$. Using vector notation, this can be written as

$$\boldsymbol{\beta} J\mathbf{s} + \delta \le 0,$$

where $\boldsymbol{\beta}_{1\times n} = 2[\beta_1, \ldots, \beta_n]$, $J_{n\times n} = I_n - \frac{1}{n}\mathbf{e}^T\mathbf{e}$, $\mathbf{e}_{1\times n} = [1, \ldots, 1]$ and $\delta = \sum_{j=1}^{n}\delta_j$.

In the following we show that the random variable $Q = \boldsymbol{\beta} J\mathbf{s} + \delta$ is normally distributed.

Proof. Assuming that the user resides at location ℓ_1 during localization, the observed RSS fingerprint $\mathbf{s} = [s_1, \ldots, s_n]^T$ is a multivariate Gaussian random vector, i.e. $\mathbf{s} \sim \mathcal{N}(\mathbf{r}_1, \Sigma)$, where $\Sigma = \sigma^2 I_n$.

We define the random vector $\mathbf{z} = [z_1, \ldots, z_n]^T$, where $z_j = s_j - \bar{s},\ j = 1, \ldots, n$ and $\bar{s} = \frac{1}{n}\sum_{k=1}^{n} s_k$. The random vector $\mathbf{z}$ is a linear function of the multivariate normal vector $\mathbf{s}$

$$\mathbf{z} = J\mathbf{s},$$

where $J = I_n - \frac{1}{n}\mathbf{e}^T\mathbf{e}$. Thus, $\mathbf{z}$ is also multivariate normal vector , i.e. $\mathbf{z} \sim \mathcal{N}(J\mathbf{r}_1, \Sigma_z)$, where $\Sigma_z = J\Sigma J^T$.

Next, the random variable Q can be viewed as a linear function of the multivariate normal $\mathbf{z}$, i.e. $Q = \boldsymbol{\beta}\mathbf{z} + \delta$. Therefore, Q is also normally distributed with mean and variance

$$\begin{aligned} \mu_Q &= \boldsymbol{\beta} J\mathbf{r}_1 + \delta \\ \sigma_Q^2 &= \boldsymbol{\beta} J\Sigma J^T\boldsymbol{\beta}^T. \end{aligned}$$

□

H.2 Derivation of the SSD Analytical Model

In the case of SSD differential fingerprints, the NN localization method will return the correct location ℓ_1 only if the following condition is satisfied

$$\check{d}_1^2 \leq \check{d}_2^2.$$

Using Theorem 2 the above condition is equivalent to

$$\begin{aligned} \check{d}_1^2 \leq \check{d}_2^2 &\Leftrightarrow d_1^2 - 2n(r_{1\rho} - s_\rho)(\bar{r}_1 - \bar{s}) + n(r_{1\rho} - s_\rho)^2 \leq d_2^2 - 2n(r_{2\rho} - s_\rho)(\bar{r}_2 - \bar{s}) + n(r_{2\rho} - s_\rho)^2 \\ &\Leftrightarrow d_1^2 - 2n(r_{1\rho}\bar{r}_1 - r_{1\rho}\bar{s} - \bar{r}_1 s_\rho) + n(r_{1\rho}^2 - 2r_{1\rho}s_\rho) \leq d_2^2 - 2n(r_{2\rho}\bar{r}_2 - r_{2\rho}\bar{s} - \bar{r}_2 s_\rho) + n(r_{2\rho}^2 - 2r_{2\rho}s_\rho) \\ &\Leftrightarrow d_1^2 - d_2^2 - 2n(r_{1\rho} - \bar{r}_1 + \bar{r}_2 - r_{2\rho})s_\rho - 2n(r_{2\rho} - r_{1\rho})\bar{s} + n(r_{1\rho}^2 - 2r_{1\rho}\bar{r}_1 - r_{2\rho}^2 + 2r_{2\rho}\bar{r}_2) \leq 0 \\ &\Leftrightarrow 2\sum_{j=1}^{n}\beta_j s_j + \sum_{j=1}^{n}(r_{aj}^2 - r_{bj}^2) - 2n(\bar{r}_2 - \bar{r}_1 - \beta_\rho)s_\rho - 2\beta_\rho\sum_{j=1}^{n}s_j + \sum_{j=1}^{n}(r_{1\rho}^2 - 2r_{1\rho}\bar{r}_1 - r_{2\rho}^2 + 2r_{2\rho}\bar{r}_2) \leq 0 \\ &\Leftrightarrow 2\sum_{j=1}^{n}(\beta_j - \beta_\rho)s_j - 2n(\frac{1}{n}\sum_{j=1}^{n}(r_{bj} - r_{aj}) - \beta_\rho)s_\rho + \sum_{j=1}^{n}(r_{aj}^2 + r_{1\rho}^2 - r_{bj}^2 - r_{2\rho}^2 - 2r_{1\rho}\bar{r}_1 + 2r_{2\rho}\bar{r}_2) \leq 0 \\ &\Leftrightarrow 2\sum_{j=1}^{n}(\beta_j - \beta_\rho)s_j - 2\sum_{j=1}^{n}(\beta_j - \beta_\rho)s_\rho + \sum_{j=1}^{n}\epsilon_j \leq 0 \\ &\Leftrightarrow 2\sum_{j=1}^{n}(\beta_j - \beta_\rho)(s_j - s_\rho) + \sum_{j=1}^{n}\epsilon_j \leq 0, \end{aligned}$$

where $\beta_j = (r_{2j} - r_{1j})$, $\beta_\rho = (r_{2\rho} - r_{1\rho})$ and $\epsilon_j = (r_{1j}^2 + r_{1\rho}^2 - r_{2j}^2 - r_{2\rho}^2 - 2r_{1\rho}\bar{r}_1 + 2r_{2\rho}\bar{r}_2)$.