OUKA
Osaka University Knowledge Archive

Title	Mobile Node Localization Focusing on Human Behavior in Pedestrian Crowds
Author(s)	樋口, 雄大
Citation	大阪大学, 2014, 博士論文
Version Type	VoR
URL	https://doi.org/10.18910/34572
rights	
Note	

Osaka University Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

Osaka University

Mobile Node Localization Focusing on Human Behavior in Pedestrian Crowds

Submitted to
Graduate School of Information Science and Technology
Osaka University

January 2014

Takamasa HIGUCHI

List of Publications

Related Journal Articles

1. Takamasa Higuchi, Sae Fujii, Hirozumi Yamaguchi, and Teruo Higashino: "Efficient Localization for Intermittently Moving Nodes," *IPSJ Journal*, vol. 52, no. 1, pp.197–208, January 2011. (in Japanese)

2. Takamasa Higuchi, Sae Fujii, Hirozumi Yamaguchi, and Teruo Higashino: "Mobile Node Localization Focusing on Stop-and-Go Behavior of Indoor Pedestrians," *IEEE Transactions on Mobile Computing*, 2013. (in press)

3. Takamasa Higuchi, Hirozumi Yamaguchi, and Teruo Higashino: "Relative Position Estimation Using Dead Reckoning and Received Signal Strength of Bluetooth," *IPSJ Journal*, vol. 54, no. 8, pp.2048–2060, August 2013. (in Japanese)

4. Takamasa Higuchi, Hirozumi Yamaguchi, and Teruo Higashino: "Context-Supported Local Crowd Mapping via Collaborative Sensing with Mobile Phones," *Pervasive and Mobile Computing*, *Elsevier*, 2013. (in press)

Related Conference Papers

1. Takamasa Higuchi, Sae Fujii, Hirozumi Yamaguchi, and Teruo Higashino: "An Efficient Localization Algorithm Focusing on Stop-and-Go Behavior of Mobile Nodes," in *Proceedings of the 9th IEEE International Conference on Pervasive Computing and Communications (PerCom '11)*, pp.205–212, March 2011.

2. Takamasa Higuchi, Hirozumi Yamaguchi, and Teruo Higashino: "Clearing a Crowd: Context-Supported Neighbor Positioning for People-Centric Navigation," in *Proceedings of the 10th International Conference on Pervasive Computing (Pervasive '12)*, pp.325–342, June 2012.

Other Conference Papers

1. Hirozumi Yamaguchi, Takamasa Higuchi, and Teruo Higashino: "Collaborative Indoor Positioning of Mobile Nodes," in *Proceedings of the 6th International Con-*

ference on Mobile Computing and Ubiquitous Networking (ICMU '12), pp.156-163, May 2012.

2. Masaki Inokuchi, Takamasa Higuchi, Hirozumi Yamaguchi, and Teruo Higashino: "Autonomous Recognition of Emergency Site by Wearable Sensors," in *Proceedings of the 10th IEEE International Conference on Cyber, Physical and Social Computing (CPSCom '12)*, pp.400–409, November 2012.

3. Yusuke Wada, Takamasa Higuchi, Hirozumi Yamaguchi, and Teruo Higashino: "Accurate Positioning of Mobile Phones in a Crowd using Laser Range Scanners," in *Proceedings of the 9th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '13)*, pp.441-446, October 2013.

Abstract

With a variety of sensors being available in off-the-shelf mobile devices, pervasive computing systems have become able to capture the situations that users are currently involved (*e.g.*, location, activity and social context) to provide various people-centric services. Since human activity is closely dependent on one's location, positioning faculty is usually an essential building block of such situation-aware applications. While GPS has been widely used as standard localization technology for mobile devices, a number of emerging applications require detailed user location information indoors where GPS hardly works.

Despite considerable research effort to develop universal indoor positioning systems, unfortunately, it is still an open problem. A major difficulty in spreading indoor localization systems arises from the trade-off between accuracy and their costs in terms of infrastructure deployment and effort for calibration. Although infrastructure-based localization systems using ultrasound, infra-red or radio signals can provide high position resolution, they usually need a huge number of embedded sensors (*i.e.*, anchors) on the walls or ceilings. Fingerprint-based localization may address such hardware-related problems using environmental signatures at each location. However, they incur considerable calibration effort for collecting reference signatures over the whole building. While pedestrian dead reckoning (PDR) technology has been investigated to provide low-cost indoor positioning solutions, it usually cannot stand alone to provide sufficient positioning accuracy due to sensor noise, unexpected human motion and other environmental factors. Thus, to provide fine-grained position information to people indoors is still a big challenge.

Now that smartphones are becoming pervasive and many people participate in some location-dependent services, they would share a common demand for finer-grained positioning. It motivates us towards another potential solution; collaborating with other users to achieve better accuracy, while reducing the dependence on infrastructure and the initial calibration effort. In order to maximize effectiveness of the collaboration, we focus on common human behavior in pedestrian crowds. People in indoor environments (*e.g.*, in a shopping mall or an exhibition hall) usually move around the area of interest in a "stop-and-go" manner. In crowded situations like exhibitions and parties, people often move together with some others, dynamically forming a "group." Effectively incorporating such common characteristics of human behavior into the algorithm design, we aim to cope with the problems in the existing localization systems. In this dissertation, such a new category of localization algorithms will be termed *mobility-aware cooperative localization*, and three primary contributions will be made toward this direction.

Firstly, we design an energy-efficient cooperative localization algorithm for mobile devices, which achieves accurate tracking of mobile devices with a reduced number of anchors. We assume that at least three anchors are deployed at known locations in the target area, and all the nodes have ad hoc communication and range measurement faculties to enable accurate peer-to-peer distance measurement. The key idea is to collaboratively find neighboring mobile devices that are temporarily stopping at the current locations, and use their estimated positions to complement a small number of anchors. Battery consumption of the mobile devices is also optimized by automatically adjusting the position update intervals according to their movement state. Experimental results show that the proposed method achieves the average localization error of 0.2m, which is much lower than a conventional cooperative approach. Also, it reduces the localization frequency by up to 77%, while achieving equivalent tracking performance to the conventional method.

Secondly, we design a general framework for performance analysis of the mobility-aware cooperative localization algorithms that exclude the nodes in motion from the set of reference points. In order to clarify when and how they can achieve optimal performance, we derive the theoretical lower bound of the localization errors. Through a case study, we compare performance of the proposed localization algorithm above with the theoretical bound, and show that the errors asymptotically approach the lower bound as more than 50% of neighboring devices maintain correct movement state. The framework of error analysis is also applicable to any other cooperative localization algorithms that select pseudo-anchors based on their movement state. As well as the error analysis, performance of the mobility-aware approaches is also analyzed from various aspects, seeking strategies to maximize their effectiveness. Based on extensive simulations and experiments using Android smartphones, we show several important observations regarding anchor deployment strategy, effectiveness of accelerometer-based motion detection and combination with PDR-based trajectory estimation, etc.

Thirdly, the notion of mobility-aware cooperative localization is extended for infrastructure-free localization for mobile devices. Assuming the situations where anchor deployment and accurate peer-to-peer distance measurement are not available, our goal here is to provide pedestrians' position information with reasonable accuracy using only built-in functions of commercial mobile devices (*e.g.*, smartphones). To support emerging people-centric applications such as mobile social navigation, we design a positioning system called *PCN*, which provides relative positions of nearby mobile phone users. PCN effectively combines trajectory estimation by PDR and proximity sensing based on received signal strength (RSS) of Bluetooth radio. By overlapping the estimated traces at the points where the users seem to have encountered, PCN derives the relative position between the users. Utilizing the feature of "group activity," it reduces the effect of sensor noise and other error-inducing factors. Through a field experiment using Android smartphones, we show that the proposed method successfully enhances positioning accuracy by 28% (from 4.16m to 3.01m). The error correction mechanism of PCN is also applicable to absolute positioning based on PDR or RSS-based peer-to-peer distance measurement.

Through these contributions, it will be shown that the mobility-aware neighbor collaboration mechanism offers improved cost/accuracy trade-offs. By combining the two mobility-aware approaches above according to the accuracy requirement and the cost limitations, we could strongly support a variety of people-centric applications that require accurate position information of mobile devices in a ubiquitous manner.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent advances in personal sensing technology have opened up a new stage of mobile applications. With a variety of sensors being available in off-the-shelf mobile phones, mobile systems can capture the situations that users are currently involved (*e.g.*, location, activity and social context) to provide various people-centric services. Since human activity is closely dependent on one's location, positioning faculty is usually an essential building block of such applications. When the application requires only district-level granularity allowing a typical error of a few hundred meters, cellular-based localization [2, 3, 4] would serve the purpose. If necessary, GPS usually provides finer accuracy that can identify the building where the user is located.

While GPS has been widely used as standard localization technology for mobile devices, a number of emerging applications, including indoor pedestrian navigation [5, 6], context sensing [7, 8] and activity analysis [9, 10], require more detailed user location information inside the building with floor-level, room-level or spot-level granularity. As an example, let us consider trajectory analysis in an exhibition hall. In such situations, history of booths where the user has visited would provide rich information for context-aware applications like life-logging [11]. The system could also suggest items in which the user is potentially interested by analyzing her trace in the exhibition hall.

As well as focusing on location and situation of individual users, some emerging services aim to support interaction with other people who have social relationship with the users themselves [12, 13]. For instance, let us think of a party place as shown in Fig. 1.1. Since the place is highly crowded with those present, our view is often obstructed by the surrounding people as Fig. 1.2. Thus we can hardly find a particular person in such a crowd even if we know that he/she is nearby. Mobile social navigation would help us in such situations by guiding the user to the friend she is looking for. These applications also require fine-grained position information of the users themselves and/or relative position to the surrounding people with a few meters accuracy in indoor environments where GPS rarely works.

Despite considerable research effort to develop universal indoor positioning systems, unfortunately, it is still an open problem. A major difficulty in spreading indoor localization systems arises from the trade-off between accuracy and their costs (in terms of

Figure 1.1: A scene of a party

Figure 1.2: Finding a person in a crowd

infrastructure deployment and effort for data collection in a training phase). Although infrastructure-based localization systems using ultrasound [14, 15], infra-red [16, 17, 18] or RF [19, 20, 21] can provide high position resolution, they usually need a huge number of embedded sensors (*i.e.*, anchors) on the walls or ceilings. Fingerprint-based localization technology may partially address such hardware-related problems by using environmental signatures at each location [22, 23, 24, 25, 26]. However, they incur considerable initial effort for collecting reference signatures over the whole building. In addition, the environmental signatures can change over time. Consequently, the localization accuracy can gradually decline unless reference signatures are updated at certain time intervals. Pedestrian dead reckoning (PDR) [6, 27, 28] estimates trajectories of the pedestrians using inertial sensors (*e.g.*, accelerometers, compasses and gyro sensors) in their mobile devices. Although it enables self-localization of commercial mobile devices without relying on any infrastructure or preliminary data collection, it cannot stand alone to provide reasonable positioning accuracy since errors are rapidly accumulated in the estimated traces due to sensor noise and unexpected human motion. Thus, to provide fine-grained position information to people indoors, *e.g.,* exhibition patrons, visitors of museums, and customers at shopping malls is still a big challenge.

Now that smartphones are becoming pervasive and many people participate in some location-dependent services (while they may run different sets of applications), they would share a common demand for finer-grained positioning. It motivates us towards another potential solution; collaborating with other users to achieve better accuracy, while reducing the dependence on infrastructure and the initial calibration effort. The goal of this dissertation is to embody this idea for providing a reasonable solution toward low-cost and energy-efficient indoor pedestrian localization.

Originally, such neighbor collaboration mechanisms have been investigated for self-localization in wireless sensor networks (WSNs) [29, 30, 31, 32, 33]. In order to estimate locations of the sensor nodes that are scattered in the environment, each node exchanges wireless beacons with its neighbors to estimate their distance. Some of the nodes are equipped with GPS receivers and serve as anchor nodes, providing accurate location information of themselves. If a node can obtain the position and distance information from a sufficient number of anchors, then it can estimate its location. Otherwise, the node also utilizes estimated positions of the neighboring non-anchor nodes as reference points for localization (*i.e.*, pseudo-anchors). Thus it effectively complements a small number of anchors and achieves reasonable positioning accuracy. One may think that the mechanism of cooperative localization can be directly applied to localization of mobile devices. However, in the mobile environment it suffers from the following dilemma: To accomplish acceptable accuracy in cooperative localization of *mobile nodes*, frequency of position updates should be sufficiently high. Otherwise, the position error of mobile nodes will be accumulated as they move, which seriously degrades the estimation accuracy of other nodes when the nodes are used as reference points. On the other hand, there is a conflicting demand that the localization frequency should be reduced in terms of saving battery and network resources because the resource consumption generally increases with the total number of localization attempts. Since most existing cooperative localization algorithms are designed for stationary sensor nodes, basically they do not care about localization timing and frequency. Although some methods such as [34, 35, 36, 37] are designed for cooperative localization in mobile environment, to the best of our knowledge, none of them tackles these problems.

Collaboration with neighboring devices would also be a promising approach to reducing trace estimation errors with PDR. The first work in this direction was carried out in [38], in which the PDR errors are mitigated by detecting proximity between the devices via wireless ad-hoc communication. Whenever two users come close within the radio range of Bluetooth, their traces are adjusted so that their current position estimates become the same location. While it can effectively reduce the errors in PDR traces, the average position error is still a few tens of meters due to the limited granularity of proximity sensing.

In order to provide a reasonable cooperative localization solution for mobile devices, we focus on common human behavior in pedestrian crowds. Existing cooperative localization algorithms assume that (i) all the nodes are completely stationary (as in [29, 30, 31, 32,

33]), or (ii) all the nodes in the environment are mobile and may move continuously and independently [34, 37]. Considering the pedestrians' behavior in practical situations (*e.g.*, in an exhibition hall or in a shopping mall), both assumptions would be extreme and overlook some characteristics of human mobility. In a shopping mall, customers often stop at shelves to choose the items they will buy. Visitors in an exhibition hall would also stay at several booths to see the displayed content. Thus people usually move around the area of interest in a "stop-and-go" manner, rather than continuously moving or being stationary. In addition, in crowded situations like exhibitions and parties, people naturally move together with some others, forming a "group" in a dynamic fashion. Thus it would often happen that a group of multiple persons move in a similar way. Effectively incorporating such common characteristics of human behavior into the algorithm design, we aim to cope with the problems in the existing cooperative localization systems. In this dissertation, such a new category of localization algorithms will be termed *mobility-aware cooperative localization*, and three primary contributions will be made toward this direction.

Firstly, we design an energy-efficient cooperative localization algorithm for mobile devices, which copes with the challenges in existing cooperative approaches above. We assume that at least three anchors are deployed at known locations in the target area, and all the nodes have ad hoc communication and range measurement faculties (*e.g.*, ultrasound transducers) to enable accurate peer-to-peer distance measurement. To pursue the best trade-off between localization frequency and positioning accuracy, it focuses on the stop-and-go behavior of mobile nodes, which is generally followed by pedestrians indoors. If a node has moved after its last localization round, then its estimated position may contain an unbounded error. On the other hand, if it is sure that the node has stayed at the current location since the last localization attempt, its position error is expected to be within a reasonable range even if the estimated position has not been updated for a long period of time. The basic idea of the proposed method is to collaboratively find a movement state (*moving* or *static*) of each mobile node, and select only *static* nodes as reference points for localization. Thus it effectively prevents the error propagation from the nodes in motion. The state information is also helpful to enhance the energy efficiency, since nodes in *static* state need not update their estimated positions until they move again. The state estimation is completed in a localization process using only distance information between the nodes, which is inherently essential for distance-based localization. The experimental results show that the average error of the proposed method is about 0.2m, which is much lower than a conventional cooperative approach that uses all the neighboring nodes as reference points for localization regardless of their movement state. Also, the proposed method could reduce the localization frequency by up to 77% to achieve the similar tracking performance to the conventional method, which repeatedly performs localization at constant time intervals.

Secondly, we design a general framework for performance analysis of the mobility-aware cooperative localization algorithms that discriminate pedestrians' movement states. While the proposed method in the first contribution employs a collaborative movement

detection mechanism based on peer-to-peer distance measurement, of course there would be other potential approaches for the movement state detection. For example, some of the existing cooperative localization algorithms maintain confidence of estimated positions to mitigate error propagation between the neighboring nodes [33, 39]. Such confidence values may serve as a reasonable indicator to find their movement state. Motion detection using accelerometers in mobile devices may also be a possible alternative solution. In order to discuss optimality of such localization algorithms, we need some criteria that appropriately take error-inducing factors (*e.g.*, distance measurement noise) into account. For that purpose, we derive the theoretical lower bound of localization errors when the nodes in motion are excluded from the set of reference points, and show how it can be applied to the performance analysis of specific localization algorithms. Through a case study, we compare performance of the proposed localization algorithm above with the theoretical bound. The results show that localization errors of the proposed method asymptotically approach the theoretical lower bound as more than 50% of neighboring devices maintain correct movement state. It means that the proposed method provides optimal performance in most practical situations. The error analysis framework is general in nature and can be also applied to any other cooperative localization algorithms that select pseudo-anchors based on their movement state. As well as the error analysis, performance of the mobility-aware approaches is also analyzed from various aspects (*e.g.*, resource consumption, robustness against node density and range measurement errors), seeking strategies to maximize their effectiveness. Based on extensive simulations and experiments using Android smartphones, we show important observations regarding anchor deployment strategy, effectiveness of accelerometer-based motion detection and combination with PDR-based trajectory estimation, etc.

Thirdly, the notion of mobility-aware cooperative localization is extended for infrastructure-free localization of mobile devices. Assuming the situations where anchor deployment and accurate peer-to-peer distance measurement are not available, our goal here is to provide pedestrians' position information with reasonable accuracy using only built-in functions of commercial mobile devices (*e.g.*, smartphones). To support emerging people-centric applications such as mobile social navigation, we design another novel positioning system called *PCN* (the acronym of people-centric navigation), which provides relative positions of nearby mobile phone users to create a *local map of surrounding crowd*. It employs trajectory estimation by PDR and proximity sensing based on received signal strength (RSS) of Bluetooth radios, both of which can be accomplished by off-the-shelf mobile phones. By overlapping the estimated traces at the points where the users seem to have encountered, PCN derives the relative position between the users. A challenge for this approach is dealing with large position errors due to sensor noise, variance of Bluetooth RSS and other environmental factors. Position errors in the PDR traces may grow up to tens of meters, which would seriously degrade the relative position accuracy. Furthermore, different Bluetooth RSS values can be observed at the same distance due to multipath effect and human presence, which confuses proximity sensing. PCN copes with the problems

by focusing on *collective activity* of the people in the crowd. In crowded situations like exhibitions and parties, people often move together with some others, forming a "group." The groups may be formed by friends, families, colleagues, or even strangers who are just moving toward the same direction, and thus can dynamically change with time. PCN captures similarity of their activities by gathering mobile phone's acceleration, direction and Bluetooth RSS, and then corrects deviation of the estimated traces by harmonizing with the traces of other group members. Such group-based error correction alleviates the impact of unstableness in the heading estimation, misdetection of user's steps and variance of Bluetooth RSS, which may affect the relative position accuracy. Through a field experiment in a public trade fair, it has been shown that PCN achieves median relative position accuracy of 3.01m, which would be sufficient for mobile social navigation or similar people-centric applications. The error correction mechanism of PCN is also applicable to absolute positioning based on PDR or RSS-based peer-to-peer distance measurement.

Through these contributions, it will be shown that the mobility-aware neighbor collaboration mechanism offers improved cost/accuracy trade-offs by effectively enhancing the quality of location sensing without relying on additional infrastructure deployment or heavy calibration effort. If anchor devices can be sparsely deployed and mobile devices are capable of accurate peer-to-peer distance measurement, the proposed method in the first contribution can achieve sub-meter positioning accuracy. Otherwise PCN in the third contribution would become an alternative solution, providing a few meters accuracy using only off-the-shelf mobile phones. By combining these two approaches according to the accuracy requirement and the cost limitations, we could strongly support a variety of people-centric applications that require accurate position information of mobile devices in a ubiquitous manner.

The rest of this dissertation is organized as follows. Chapter 2 reviews related work on indoor pedestrian tracking. Chapter 3 describes the design and performance of the cooperative localization algorithm focusing on stop-and-go behavior of indoor pedestrians. Chapter 4 clarifies the effectiveness of mobility-aware cooperative localization through theoretical analysis and extensive simulations. Chapter 5 explains the detailed design of PCN, followed by performance evaluation through simulations and field experiments using Android smartphones. Finally, Chapter 6 summarizes and concludes this dissertation.

Chapter 2

Related Work

2.1 Infrastructure-based Localization

2.1.1 Anchor-based Indoor Positioning Systems

A number of methods have been investigated to estimate location of mobile devices. The most fundamental, but robust approach would be infrastructure-based localization. RFID tags are widely used for indoor object/human tracking [16, 17, 18] and some of them have been on the market. RFID readers are generally placed at strategic positions to detect tags that pass in their read range. Thus, localization accuracy of the RFID-based localization systems basically depends on the number of readers in the environment.

Cricket [14] and Active Bat [15] are well-known systems that employ TDoA (time difference of arrival) of ultrasound and radio signals for distance measurement between mobile devices and fixed beacon devices (*i.e.*, anchors). Accurate range measurement by TDoA offers high position resolution with a typical error of a few centimeters. UWB (Ultra-Wide Band) radio-based distance measurement techniques [19, 20, 21] also provide sub-meter accuracy using propagation delay of radio signals. Despite their excellent accuracy and robustness, these methods have a common issue in terms of cost for installation and maintenance; to achieve accurate positioning, they typically need dense anchor deployment with a few meters spacing, which severely limits their scalability.

2.1.2 Fingerprint-based Localization

Fingerprint-based localization has also been well examined as another major approach for indoor positioning. In particular, WiFi-based systems [23, 24, 25, 26, 40] are fairly popular and widely used for localization of mobile devices. In a learning phase, RSS of radio beacons from multiple WiFi access points are collected at each location in the building to construct a "radio map." Once the learning is completed, location of mobile devices can be identified by matching the observed radio signature with those in the radio map. Furthermore, the recent trend of configuration-free localization mitigates effort in the calibration process [24, 41]. Ecolocation [42] employs the ordered sequence of received signal strength measurements taken from multiple access points to enhance robustness

against signal fluctuations. As well as WiFi RSS, ambient sound [43], GSM [4, 44] and FM broadcast radios [45, 46] have also been proved to serve as effective location signatures in indoor environments. SurroundSense [22] employs environmental signatures like ambient sound, acceleration, color and light, which can be sensed by cameras and microphones in mobile phones. Combining these optical, acoustic and motion attributes with WiFi signatures, it can robustly distinguish adjacent locations that are separated only by a wall. While most of such fingerprint-based localization systems are designed to work with off-the-shelf mobile devices, the learning phase to collect reference signatures usually incurs heavy effort. Furthermore, both radio and environmental signatures can change over time. Consequently, the localization accuracy can gradually decline unless reference signatures are updated at regular intervals.

2.1.3 Device-free Passive Localization

The localization systems in the previous sections assume that targets (*i.e.*, pedestrians) have mobile devices with range measurement and/or wireless communication capability. In contrast, there is another class of localization systems called *device-free localization*, which do not require people to carry any wireless devices [47]. The basic idea underpinning these systems is that the presence of pedestrians should cause variations of some physical quantities in the environment (*e.g.,* received signal strength and multipath components of radio signals). By capturing such changes with various sensors, they estimate current location of the pedestrians.

Some methods like [48, 49] utilize RSS values that are observed between multiple WiFi access points. Since the temporal variation in the RSS values would be mainly caused by movement of the pedestrians, their presence can be detected by comparing the current, short-term signal behavior with the average behavior over a longer period of time. They calculate moving averages or moving variances of the RSS values with two different window sizes, and detect a target if their difference becomes larger than a pre-defined threshold. Wilson et al. [50] employ a Kalman filter to improve tracking performance of such RSS-based passive localization systems. UWB radio signals have also been successfully used for device-free localization [51, 52]. While UWB receivers are more expensive than the ordinary narrowband wireless devices, they can precisely measure signal amplitudes, temporal delays and phases of each multipath component. Thus they usually achieve better positioning accuracy than the RSS-based systems.

Passive infrared (IR) sensors have also been widely used as a presence trigger for automatic light switching and surveillance systems. Since they are low-cost, low-power, and providing a reliable indication of human presence, they have been considered suitable for WSN-based pedestrian tracking systems [53, 54, 55]. For example, Zappi et al. [55] propose a human tracking system that uses a set of wireless nodes equipped with passive IR sensors. In order to cover the area of interest in a collaborative manner, the sensors autonomously form clusters consisting of two nodes. Then each cluster roughly estimates direction of pedestrians' movement and their relative positions with respect to the sen-

sors themselves. These data are periodically collected to a centralized server to estimate positions of the targets.

Crowd tracking systems using laser range scanners (LRSs) have also been well investigated owing to their ease of installation and the privacy-preserving feature. An LRS periodically transmits infrared laser pulses toward a fan-shaped area with a signal transmission range of a few tens of meters, and then receives reflected signals from the surrounding objects. Thus distance to those objects can be estimated with a few centimeters accuracy based on the time of flight of the measurement signals. The simplest approach to the LRS-based pedestrian tracking is to set the sensors at the average height of pedestrian's waist and employ distance-differencing techniques to detect pedestrians [56]. A problem is that people behind other pedestrians, obstacles or walls cannot be detected (*i.e.*, occlusion). Zhao et al. [57] mitigate such occlusion problem by setting the LRSs at the height of pedestrians' ankles.

Vision-based pedestrian tracking have attracted an extensive amount of interest from computer vision community. Cameras are usually deployed on the ceilings, watching down the area of interest to capture heads, faces or bodies of the pedestrians. A number of techniques have been proposed in terms of features, models and general architectures [58]. Some literatures like [59, 60] propose Monte-Carlo-based approaches that simultaneously detect and track multiple pedestrians in crowded scenes. Giebel et al. [61] enhance the crowd tracking performance by combining human detection using spatio-temporal shape models and the particle-filter-based Bayesian tracking. Smith et al. [62] employ Markov chain Monte Carlo optimization to handle the situations where people often enter and leave the area of interest. More recently, Fleuret et al. [63] substantially improve the position estimation accuracy and robustness against occlusions by effectively integrating images from multiple cameras.

Most of the passive localization systems above aim to detect presence and/or trajectories of the pedestrians to monitor current situation in the environment (*e.g.*, for surveillance). Therefore the position information provided by these systems is not associated with any particular persons. However, such a set of *anonymous* traces may not useful for the mobile phone users who need to know *their own positions* to employ location-based services (*e.g.*, pedestrian navigation). In order to support such applications, we need *identified location* of each mobile phone user in a crowd. In that sense, utilizing the capability of user's mobile devices would be more direct approach toward our goal.

2.2 Localization with Less Dependence on Infrastructure

2.2.1 Ad Hoc Network-based Localization

To instantly achieve indoor positioning without relying on infrastructure or war-driving, mobile ad-hoc communication-based localization may offer a reasonable option. Some methods like [64, 65, 66] utilize hop-counting techniques to roughly estimate distance to the anchors. The anchors periodically announce their location through the multihop net-

work so that each mobile node can collect the minimum number of hops to each anchor. SISR [67] enhances the error tolerance of such methods by introducing an improved residual function for the least squares method. Although they can provide rough position information without any dedicated faculties, their accuracy significantly depends on network topology. Furthermore, delay in collecting the network information severely limits the frequency of position updates, so that the tracking performance cannot be guaranteed.

In contrast, algorithms like [34, 35, 36, 37] are designed for real-time tracking of mobile nodes based on wireless connectivity between the devices and their movement constraints. TRADE [34] achieves real-time localization in a fully distributed manner. Mobile nodes estimate and update their trajectory based on neighboring nodes' trajectory information and wireless connectivity information. Záruba et al. [68] propose a distributed algorithm based on Sequential Monte Carlo method, in which the estimated position is maintained in the form of a probability distribution that is represented by a collection of sample points. Uncertainty of the estimated position due to noisy measurement is estimated from variance of the sample points and utilized to mitigate error propagation. WMCL [37] also proposes a Sequential Monte Carlo-based algorithm, which uses estimated positions of neighboring nodes to reduce computational cost and to improve accuracy. SPAWN [39] provides cooperative localization based on a belief propagation algorithm. Each node exchanges the distribution of its current position estimate (*i.e.*, a belief) and repeatedly refines its own belief using the measured distance to the neighboring nodes. Although these methods are capable of tracking mobile nodes, they commonly suffer localization frequency issue, as mentioned in Chapter 1.

2.2.2 Cooperative Localization in Wireless Sensor Networks

Cooperative localization utilizes estimated positions of neighboring nodes to complement a small number of anchors. In DOLPHIN [29], each node uses both anchors and neighboring nodes that have already been localized as reference points. Nodes can immediately estimate their positions by trilateration once they obtain distance information from a sufficient number of reference points, and this is continued until all the nodes are localized. Some algorithms are designed to mitigate position error propagation between the nodes. In [30, 31], relative confidence of the estimated position is introduced for weighting each neighbor. Cluster-based localization [32] selects spatially spread nodes to form robust quadrilaterals to improve accuracy. ILS [33] maintains uncertainty of the estimated position as well as the position itself. Nodes with high overall errors are excluded from the set of reference points to prevent errors from propagating. Since all of them are designed for stationary WSNs and thus do not consider node mobility which accumulates large position error, they cannot be directly applied to mobile scenarios.

2.2.3 Cooperative Localization for Commercial Mobile Devices

Some cooperative localization algorithms are designed for commercial mobile devices (*e.g.*, smartphones). Virtual Compass [13] employs peer-to-peer wireless communication via

Wi-Fi and Bluetooth to perform RSS-based distance measurement. Then relative position between the neighboring devices is derived on the spot using the measured distance. However, both RSS-based distance measurement and wireless connectivity information are significantly affected by multipath fading and presence of humans and obstacles in the building. In consequence, errors are typically in the range of a few meters to tens of meters, which would not be sufficient for many indoor location-based services. Chan et al. [69] propose a hybrid solution combining RSS-based peer-to-peer distance estimation with WiFi-based localization. It estimates distance to each neighbor using the pre-defined RSS-to-distance mapping model and then corrects the estimated positions obtained by a WiFi-based localization system. Considering the estimated distance to the neighboring mobile devices and confidence of their estimated positions, it generates a position correction vector that maximizes consistency with the distance information. While the collaborative error correction mechanism can mitigate the position errors, accuracy of RSS-based distance estimation is significantly affected by signal attenuation by human bodies and other obstacles in the environment, which would limit effectiveness of the position refinement in practical scenarios. Some methods like [70, 71] implement ranging systems on off-the-shelf mobile phones (*e.g.*, smartphones) using audio tones. Although they also achieve high distance resolution, the measurable range is strictly limited by the transmission range of audio tones, which is usually at most 10m with microphones and speakers of off-the-shelf mobile devices. Also they require additional effort such as device-dependent tuning and background sound noise elimination. Thus more instant and simpler positioning is preferable.

2.2.4 Pedestrian Dead Reckoning

Another approach to reducing dependence on infrastructure is to employ pedestrian dead reckoning (PDR) techniques [72, 73, 74] which estimate the walking trajectory of a person using accelerometers, digital compasses and gyro sensors. While most of previous PDR methods have assumed dedicated sensor devices attached to a specific part of a human body (*e.g.*, waist and feet), some recent works employ commercial mobile phones to support context-aware mobile applications [6, 27, 28]. However, due to sensor noise and irregular human motion, PDR cannot stand alone to obtain reasonable tracking accuracy.

An effective way to refine the trace estimation accuracy is to employ map information. In [75], errors in the PDR traces are mitigated by fusing received signal strength from WiFi access points and floor map information. CompAcc [6] extracts possible moving paths from Google Maps and matches noisy PDR traces with those reference paths to maintain reasonable tracking accuracy. Refs. [72, 74] introduce particle-filter-based approaches that match the original PDR traces with floor maps to provide accurate indoor positioning. Although these methods have proved that map matching is a powerful tool for boosting PDR accuracy, efficacy of the map-based correction would be usually limited in the situations like an event or party place where trajectories of the users are not necessarily constrained by the structure of the building.

As an alternative strategy, Kloch et al. [38] detect proximity between the devices via Bluetooth to correct PDR traces. Whenever two users come close within the radio range of Bluetooth, their traces are adjusted so that their current position estimates become the same location. The approach in [38] is the most relevant to our work in Chapter 5 in the sense that our approach also exploits proximity information to improve accuracy of PDR. While it can effectively reduce the errors in PDR traces, average position error in [38] is still a few tens of meters due to the limited granularity of proximity sensing.

Ref. [76] corrects the accumulated position error based on distance measurement to stationary anchors that are sparsely deployed in the environment. In general, positioning accuracy of such PDR-based tracking systems can be improved by increasing the number of anchors. In that sense, our method in Chapter 3 can strongly support these systems since each node can employ not only anchors but also neighboring mobile nodes to correct its accumulated PDR errors.

2.3 Energy-Efficient Localization

2.3.1 Power Saving Mechanisms for WSN-based Target Tracking

Due to the limited energy budget of sensor and mobile devices, energy efficiency is also an essential feature for localization systems. For object tracking in WSNs, typical approaches to energy conservation are to put the sensors that are not assumed to detect any target objects into a *sleep* mode, and/or to minimize the communication overhead for data collection. Tree-based approaches [77, 78, 79] construct a hierarchical tree structure among the sensor nodes to reduce redundant message transmissions upon detecting the targets. STUN [77] organizes a tree topology in which leaf nodes are the sensors for detecting moving objects. Then the sensor data are collected to a querying node (*i.e.*, a sink) located at the root of the tree. Each intermediate node maintains all the moving objects that are detected by its descendants, and sends an update message to its parent only when the set of detected objects has been changed. DCTC [78, 79] forms a tree structure called *convoy tree*, which is dynamically configured by adding and removing some nodes as the target moves. When the target first enters the detection region, sensor nodes that can detect the target collaborate with each other to select a root and construct an initial convoy tree. As the target moves, the nodes that have become far away from the target are pruned from the tree. The root also predicts the target's moving direction and activates a group of sensor nodes so that the target can be detected in a timely fashion. Dynamic-clustering-based protocols [80, 81] autonomously form clusters with neighboring nodes to facilitate collaborative data processing. When a sensor with sufficient battery and computational resources detects a target, it volunteers to act as a cluster head. Then sensors in the vicinity of the active cluster head are invited to become members of the cluster and report their sensor data to the cluster head. Thus a cluster is only formed in the vicinity of the targets. Prediction-based methods [82, 83] reduce the number of active sensors by estimating future movement of the targets. In PES [82], a sensor node that has a moving

object in its territory predicts the possible locations of the targets based on a prediction model. Then it determines a set of sensor nodes that are assumed to help tracking the target after certain period of sleeping, and sends them a wake up message. In DPR [83], upcoming locations of the targets are predicted by both sensors and a sink node based on historical mobility data. Then transmissions of sensor readings are avoided as long as the predictions are consistent with the real object movements. Although these methods have been proved to be effective for stationary WSNs, they do not consider mobility of the sensor nodes themselves and thus cannot be directly applied to mobile node localization.

2.3.2 Energy-Efficient Localization for Mobile Phones

Energy-efficient localization for mobile phones have also been investigated to support a variety of context-aware applications. A basic idea of existing solutions is strategically duty-cycling GPS to achieve a reasonable accuracy without continuously draining huge energy. EnTracked [84] employs accelerometers to distinguish movement state (*i.e.*, stationary or in-motion) of the users, while estimating user's velocity based on the estimated positions obtained by GPS. Then it combines these information to determine the timing of the next position update. EnLoc [85] interpolates between consecutive location readings from GPS by predicting user's movement based on the past mobility patterns, which contributes to reduce instantaneous error. RAPS [86] dynamically activates GPS when the expected position uncertainty exceeds a designated level and GPS fix is likely available according to the history of localization attempts at each area. All of these methods assume outdoor situations, and thus energy-efficient localization for indoor environments has yet to be investigated enough even though people spend much time indoors.

Chapter 3

An Efficient Localization Algorithm Focusing on Stop-and-Go Behavior of Indoor Pedestrians

3.1 Introduction

This chapter presents a novel cooperative localization algorithm for mobile devices, which achieves high tracking performance with reduced resource consumption. We assume that at least three anchors are deployed at known locations in the target area, and all the nodes have ad hoc communication and range measurement faculties to enable accurate peer-to-peer distance measurement. The key idea is to collaboratively find neighboring mobile devices that are temporarily stopping at the current locations, and use their estimated positions to complement a small number of anchor devices. The estimation of movement state (*i.e.*, *moving* or *static*) is completed in a localization process using only distance information between the nodes, which is inherently essential for distance-based localization. In addition, battery consumption of the mobile devices is also optimized by automatically adjusting the position update intervals according to their movement state. The experimental results show that the average error of the proposed method was about 0.2m, which is much lower than conventional cooperative localization approaches. Also, the proposed method could reduce the localization frequency by up to 77% while keeping the similar tracking performance. The efficacy is also confirmed through experiments using real human traces and a real ranging model.

The rest of this chapter is organized as follows. Section 3.2 describes the basic idea, architecture and the detailed design of the proposed localization algorithm. Sections 3.3 and 3.4 provide performance evaluation through extensive simulations and field experiments, respectively. Finally, Section 3.5 concludes this chapter.

3.2 Algorithm Design

3.2.1 Preliminaries

2-dimensional localization requires distance information from at least three reference points. Instead of obtaining distance information to many anchor nodes, the proposed method uses estimated positions of neighboring mobile nodes as reference points for localization. We will refer to such semi-mobile reference points as *pseudo-anchors*. Thus we assume that three or more anchors are deployed at known positions in a target area, and that all the nodes have ad hoc communication and range measurement faculties. For range measurement, we assume a Time Difference of Arrival (TDoA) technique using ultrasound devices. A node simultaneously transmits RF and ultrasound signals to allow a receiver node to calculate the time difference of two signals to estimate the distance. Hence, nodes are equipped with transmitters and receivers of ultrasound signals in addition to a wireless communication module. On account of its accurate ranging capability, ultrasonic ranging provides precise, robust indoor positioning, and thus has been commonly used for indoor localization systems [14, 15, 29]. Since an ultrasound transmitter has a directional range pattern, we assume that each node has several ultrasound transmitters and receivers that are arranged radially to achieve an omni-directional range pattern. Although we use ultrasonic transducers in the experiments, the proposed algorithm also works with radio-based ranging technology such as IEEE 802.15.4a or audio-based ranging systems like BeepBeep [70], which would be less affected by the directionality issue.

3.2.2 Overview

For locating a node, the proposed algorithm identifies "temporary stopping nodes" from its neighbors to select appropriate pseudo-anchors. This is also effective to reduce localization frequency, because we need not localize such stopping nodes until they move again. Based on the idea, we designed the localization algorithm as follows.

Each node A_i maintains its estimated position (x_i, y_i), speed v_i, as well as "state," which is any of *static*, *moving* and *unknown*. Anchors are stationary and thus always have *static* state, while the other nodes are initially in *unknown* state. Node A_i estimates its position, speed and state at a certain time interval, which is adapted to the estimated speed. The algorithm for interval adaption will be given in Section 3.2.5. When A_i performs localization, it simultaneously transmits RF and ultrasound signals to let its neighbors perform TDoA distance estimation. Then each neighbor A_j immediately sends back a reply message containing the estimated distance and A_j's current position to A_i, *if and only if* A_j *is in static state*. Thus A_i can locate itself using the neighbors' positions as reference points.

Note that the protocol above still implies a problem, where the nodes in *static* state may be actually moving. It may often happen since the state of a node is updated only when it is localized. To prevent such nodes that may have large position errors from being selected as pseudo-anchors, A_i verifies A_j's state based on the collected measured distance

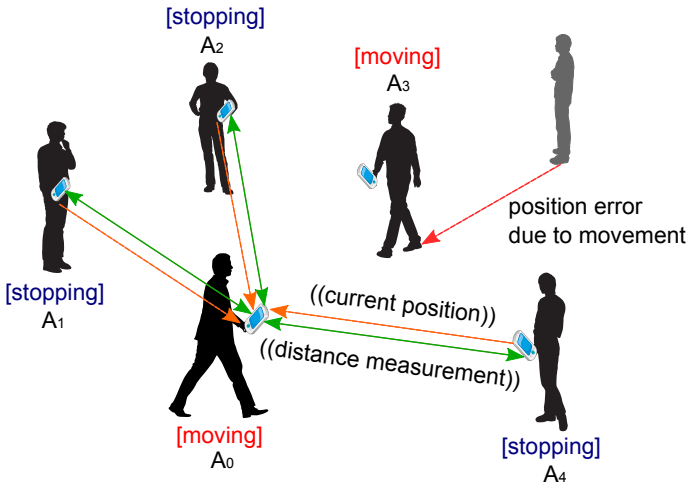


Figure 3.1: Concept of stop-and-go localization

and position information. Through a state validation process which will be described in Section 3.2.3, a neighbor A_j is selected as a pseudo-anchor if and only if it is confirmed to be actually stopping.

If three or more anchors/pseudo-anchors are found, A_i estimates its position based on the measured distance (discussed in Section 3.2.4). After that, the state of A_i is set to *static* if the estimated position is sufficiently close to the previous position, or otherwise it is set to *moving*. If only two or less *static* nodes have replied to A_i, neighbors in *moving* state also reply to A_i to let it tentatively approximate its position. In that case, the state of A_i turns into *unknown* so that it retries localization in a certain time interval.

If node A_i finds that A_j in *static* state is actually moving, A_i notifies A_j of the fact. Then A_j changes its state to *moving* and immediately attempts localization.

Fig. 3.1 illustrates a localization process of node A_0. Based on its estimated speed, A_0 starts its localization process by sending measurement signals. Then the nodes A_1, A_2 and A_4 in *static* state reply the distance and their position information to let A_0 estimate and update its position, speed and state.

3.2.3 State Validation

This section explains how to select pseudo-anchors from neighbors in *static* state in the localization process of a node A_i. Hereafter, we denote measured distance and estimated position of each neighbor A_j by $\widehat{d}_j$ and $\boldsymbol{\phi}_j = (x_j, y_j)$, respectively. Also, we represent the expected error in $\widehat{d}_j$ by σ_{r_j}, and that of $\boldsymbol{\phi}_j$ by σ_{p_j}.

If a neighbor A_j in *static* state is actually stopping, the solution space of A_i's position is assumed to be a circular ring with radius of $\widehat{d}_j$ whose center-point is at $\boldsymbol{\phi}_j$, where the width of the ring depends on the errors contained in $\widehat{d}_j$ and $\boldsymbol{\phi}_j$. The proposed method approximates the possible error range of the solution space as $\pm\epsilon_j$ where ϵ_j is given by

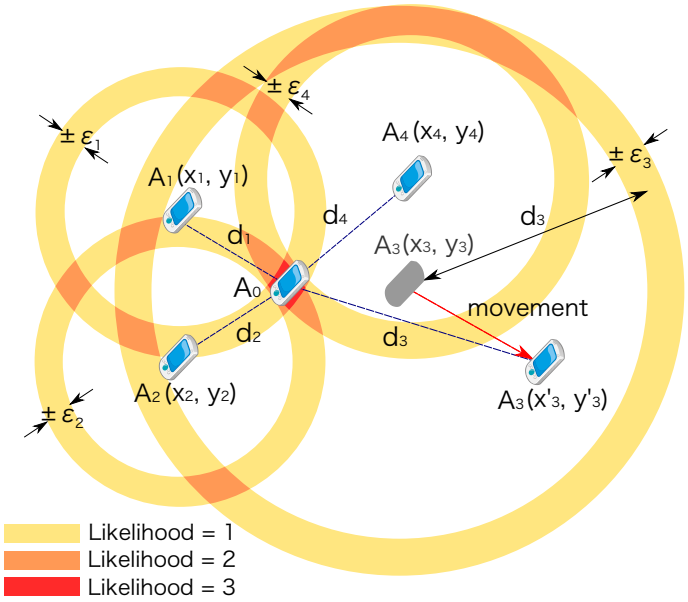


Figure 3.2: State validation (likelihood distribution)

sum of the expected errors:

$$\epsilon_j = \sigma_{r_j} + \sigma_{p_j}. \tag{3.1}$$

The position uncertainty σ_{p_j} is calculated by A_j in its localization process, and then reported to A_i along with $\widehat{d}_j$ and $\boldsymbol{\phi}_j$ as a reply message of A_j. It is estimated by Eq. (3.4), which will be explained later. Regarding σ_{r_j}, if line-of-sight (LOS) environment is assumed in the TDoA measurement (otherwise it is blocked and failed) and if the error of measured distance follows a Gaussian distribution, the error linearly increases as two nodes are distant and can be estimated as:

$$\sigma_{r_j} = \sigma_0 \widehat{d}_j \tag{3.2}$$

where σ_0 is the standard deviation of measurement errors when the distance is 1m, which can be obtained by a preliminary experiment.

Using $\boldsymbol{\phi}_j$, $\widehat{d}_j$ and ϵ_j, the circular ring of A_j can be determined. We regard that the solution is contained in the intersection of the largest number of circular rings, and choose an initial solution from the overlapped region. Then A'_j whose circular ring does not include the point is regarded as in *moving* state.

Fig. 3.2 illustrates the state validation process of A_0 where neighbors A_1, A_2, A_3 and A_4 have sent reply messages to A_0 while A_3 has actually moved. In this case, A_0 picks an initial solution from the intersection of the circular rings of A_1, A_2 and A_4. Consequently, A_3's movement is detected since the initial solution is outside the circular ring of A_3.

Based on the strategy above, we design the state validation algorithm as follows. To make the algorithm simple, we utilize a set of points to represent an area.

1. For each *static* neighbor A_j, uniformly generate candidate points in a circular ring centered at $\boldsymbol{\phi}_j$ with inner radius of $(\widehat{d}_j - \epsilon_j)$ and outer radius of $(\widehat{d}_j + \epsilon_j)$. Here we denote each candidate point by $\boldsymbol{\theta}^{(u)} = (x^{(u)}, y^{(u)})$ $(u = 1, 2, \cdots, n)$.

2. Calculate distance between $\boldsymbol{\theta}^{(u)}$ and $\boldsymbol{\phi}_j$ for each pair of u and j. We let $d_j^{(u)}$ denote the calculated distance.

3. For each candidate point $\boldsymbol{\theta}^{(u)}$, count the number of *static* neighbors A_j that satisfy $|d_j^{(u)} - \widehat{d}_j| \leq \epsilon_j$. We call this number *likelihood* of $\boldsymbol{\theta}^{(u)}$.

4. Calculate the centroid $\bar{\boldsymbol{\theta}} = (\bar{x}, \bar{y})$ of the candidate points with the largest likelihood. Then select a candidate point with the largest likelihood that is the nearest to $\bar{\boldsymbol{\theta}}$ as an initial solution.

Let $d_j^{(0)}$ denote the distance between the initial solution and $\boldsymbol{\phi}_j$. Since the initial solution is decided by a majority of *static* neighbors, $d_j^{(0)}$ should be consistent with the measured distance as long as the neighbor A_j is actually stopping. Thus we exclude from pseudo-anchors such A_j that raise inconsistency between $d_j^{(0)}$ and the original measurement $\widehat{d}_j$. More specifically, nodes A_j that do *not* satisfy $|d_j^{(0)} - d_j| \leq \epsilon_j$ are regarded as *moving* and thus excluded from the set of pseudo-anchors.

If different sets of *static* neighbors form multiple intersections of the largest number of circular rings, a wrong set of pseudo-anchors may be selected and possibly results in a large localization error. To avoid this problem, the proposed algorithm detects such failure of pseudo-anchor selection by exploiting the variance of the points in the intersection(s) from the centroid to estimate the form of the intersection(s). When n_c candidate points, say $\boldsymbol{\theta}^{(u)} = (x^{(u)}, y^{(u)})$ $(u = 1, 2, \ldots n_c)$, have the maximum likelihood, the variance is defined as

$$\sigma_c = \sqrt{\frac{\sum_{u=1}^{n_c} \{(x^{(u)} - \bar{x})^2 + (y^{(u)} - \bar{y})^2\}}{n_c - 1}}. \tag{3.3}$$

If σ_c is larger than a certain threshold (denoted by $\sigma_{c_{max}}$), the intersection is regarded as a part of multiple regions. Also we approximate the expected position error of A_i by the variance σ_c:

$$\sigma_{p_i} = \sigma_c. \tag{3.4}$$

Thus localization of A_i is accomplished only when three or more anchors/pseudo-anchors are found and the variance of the candidate points σ_c is less than $\sigma_{c_{max}}$. Otherwise, the localization fails and the state of A_i turns into *unknown*.

Let m denote the number of *static* neighbors. The total number of candidate points to be generated for localization is $\sum_{j=1}^{m} \delta_j D_j$, where δ_j is the density of points to be generated for each neighbor A_j, and D_j represents the area of the circular ring for A_j. To prevent too many points from being generated in case of large m, the proposed method adjusts δ_j as shown in Eq. (3.5). δ_{max} and δ_{min} denote the maximum value and minimum value of

δ_j, and are given as parameters. M is the limit of m, and m_a ($m_a \leq m$) is the number of anchors in the m *static* neighbors.

$$\delta_j = \begin{cases} \delta_{max} - \dfrac{m-1}{M-1}(\delta_{max} - \delta_{min}) & \text{if } m \leq M, m_a = 0 \\ \delta_{min} & \text{if } m > M, m_a = 0 \\ \delta_{max} - \dfrac{m_a - 1}{M-1}(\delta_{max} - \delta_{min}) & \text{for anchors} \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

Since anchors are definitely static, we can put confidence in location information of the anchors and thus candidate points are generated only for the anchors if $m_a > 0$. In this case, the proposed method determines δ_j for anchors according to m_a instead of m.

3.2.4 Position Estimation

After the pseudo-anchor selection, the position of A_i, say $\boldsymbol{\phi}_i$, is derived by the weighted least squares method, minimizing the following residual function:

$$D(\boldsymbol{\phi}_i) = \sum_{A_j \in \mathcal{S}} w_j (||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j|| - \widehat{d}_j)^2 \tag{3.6}$$

where $\mathcal{S}$ is the set of anchors/pseudo-anchors. The weight w_j is determined as

$$w_j = \exp(-\gamma \epsilon_j) \tag{3.7}$$

where the coefficient γ is set to 3.0.

3.2.5 Localization Intervals

Static nodes can reduce localization frequency without degrading position accuracy of themselves and their neighbors. On the other hand, nodes moving at high speed should frequently perform localization to keep their position accuracy. In the proposed method, each node adjusts its localization interval based on its speed and state to totally achieve efficient localization.

Each node estimates its speed when it performs localization. Let $\boldsymbol{\phi}_i(t')$ and $\boldsymbol{\phi}_i(t)$ denote temporally contiguous estimated positions at time t' and time t ($t' \leq t$), respectively. The speed v_i of A_i is then estimated as

$$v_i = \frac{||\boldsymbol{\phi}_i(t) - \boldsymbol{\phi}_i(t')||}{t - t'} \tag{3.8}$$

If this speed is substantially low, A_i is expected to be stopping. Specifically, A_i is regarded to be stopping and v_i is set to zero when $||\boldsymbol{\phi}_i(t) - \boldsymbol{\phi}_i(t')||$ is less than $\max(\sigma_{p_i}(t), \sigma_{p_i}(t'))$, where $\sigma_{p_i}(t)$ and $\sigma_{p_i}(t')$ are the expected errors of $\boldsymbol{\phi}_i(t)$ and $\boldsymbol{\phi}_i(t')$, respectively. Otherwise, A_i is regarded to be moving and it schedules the next localization in $I_v(v_i)$ seconds. $I_v(v_i)$ is updated in each localization process as

$$I_v(v_i) = \min \left\{ I_{v_{max}}, \frac{1}{cv_i} \right\} \qquad (v_i > 0) \tag{3.9}$$

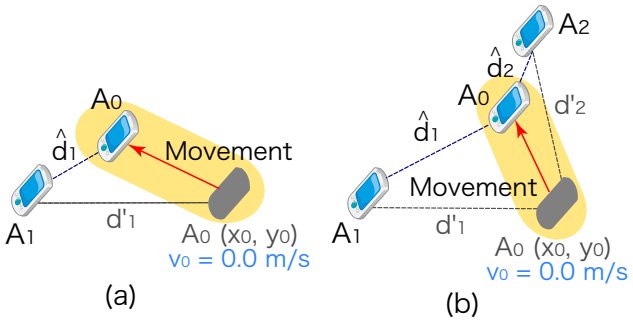


Figure 3.3: Movement detection by multiple neighbors

where c is the number of localization attempts per traveled distance, which is determined according to the required tracking resolution. We let $I_v(v_i)$ be less than a certain level (denoted by $I_{v_{max}}$) so that nodes that have just started moving or those moving at low speed can perform localization at appropriate intervals.

On the other hand, it is desirable that nodes do not perform localization while they are stopping. While a node is in *static*, the state is repeatedly verified by its neighbors when they are referred as a pseudo-anchor, and thus its movement would be immediately detected and notified. Therefore, once the state of a node turns into *static*, position updates are suspended until it receives a movement notification message from its neighbor. Exceptionally, *static* nodes voluntarily perform localization to verify their state by themselves when they have not received range measurement signals for a long period of time (*i.e.*, I_s seconds).

We note that the movement detection by a neighbor may sometimes fail depending on the direction of the movement. For example, let us think of detecting the movement of A_0 in Fig. 3.3. Let $\widehat{d}_1$ denote the measured distance between A_0 and a neighbor A_1, and let d'_1 denote the distance between their estimated positions. If the difference between $\widehat{d}_1$ and d'_1 is sufficiently large as shown in Fig. 3.3 (a), the movement of A_0 can be probably detected by A_1. However, $|\widehat{d}_1 - d'_1|$ is not always large as shown in Fig. 3.3 (b) even if the traveled distance of A_0 is large. In this case, A_1 may miss the movement of A_0 and may improperly select it as a pseudo-anchor in its localization process. Fortunately it may not greatly affect the localization accuracy of A_1 as far as $\widehat{d}_1$ and d'_1 are consistent. Also, we could expect that the movement of A_0 would be detected soon by other neighbor in different direction, such as A_2 in Fig. 3.3 (b).

Finally, we design the localization interval in the case that the localization has failed. When a node cannot find three ore more anchors/pseudo-anchors (or cannot determine the position uniquely), it tentatively estimates the position using the estimated positions of *moving* neighbors as well as those of *static* ones. Then it retries localization in a time controlled by the number of consecutive failures. If the node fails localization F times consecutively, it schedules the next localization in FT_f ($F \leq F_{max}$) seconds where T_f is constant time duration to lessen localization frequency when the nodes cannot find proper

anchors.

3.2.6 Protocol Design

Since each node autonomously determines the timing of localization, in the proposed algorithm, more than one node may attempt localization simultaneously. If those nodes conduct TDoA measurement at the same time, the range measurements probably fail. To cope with this problem, we design a collision avoidance mechanism similar to the RTS/CTS mechanism in CSMA/CA-based protocols.

When a node performs localization, it broadcasts a *Request To Measure (RTM)* message before sending TDoA measurement signals, and occupies the bandwidth for measurement signals for a while. Also, each node has a timer to maintain the time when the bandwidth for measurement signals is occupied by other nodes. We call the time *Network Allocation Vector (NAV)*. Whenever a node receives an RTM message, it sets the NAV timer to T_{cycle} seconds, where T_{cycle} is determined considering the maximum propagation time of ultrasound. Then it decrements the NAV timer over time. While the NAV timer is more than zero, a node postpones its localization.

The nodes that have delayed localization can perform it when the NAV timer reaches zero. To prevent the same node from being delayed for a long time, it lets each node wait for backoff time before sending an RTM message. The backoff time $T_{backoff}$ is determined such that a node that has been delayed for longer time can have shorter backoff time using the following formula;

$$T_{backoff} = CW \cdot \exp(-at) \tag{3.10}$$

where t is the delay time, CW is the maximum backoff time and a is a constant parameter to define characteristics of the backoff time. Larger CW can reduce collision probability of RTM messages, but requires longer time for each localization round.

The transmission range R of RTM messages should be larger than $2r$ where r is that of TDoA measurement signals. Thus nodes that are more than R away from each other can perform localization at the same time without causing collision of the measurement signals.

3.3 Evaluation

3.3.1 Simulation Settings

We have evaluated the performance of the proposed method through several simulations using the network simulator QualNet [87]. We assume a 15m $\times$ 15m field where 4 anchors are installed as shown in Fig. 3.4. Mobile nodes follow the following realistic mobility model that models the behavior of people looking around indoor space such as stores and museums. We divide the field into 3$\times$3 cells and assume that nodes probabilistically determine their destination cells on each occasion of movement. In general, people in museums and stores tend to move into nearby points of interest. Based on the observation, we select

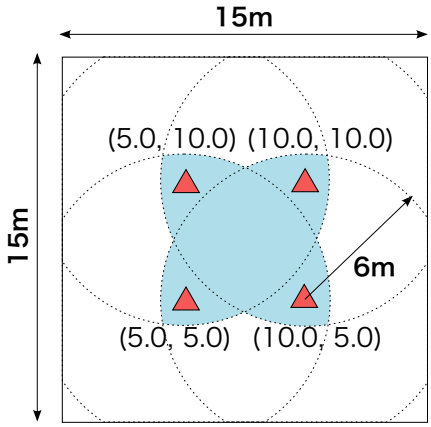


Figure 3.4: Field map

Table 3.1: Parameter settings

coefficient in determining localization intervals (c)	0.80
maximum localization interval for *moving* nodes ($I_{v_{max}}$)	3.0 seconds
localization interval for *static* nodes (I_s)	5.0 seconds
unit localization interval on failures (T_f)	1.0 seconds
maximum localization interval on failures ($F_{max}T_f$)	5.0 seconds
coefficient in determining backoff time (a)	0.76
tolerable position error ($\sigma_{c_{max}}$)	1.0 m

neighboring cells as their destination with relatively high probability of 0.7 while distant cells are picked out with probability of 0.3. Each node travels to a randomly selected point in the destination cell with probability p or stop for 10 seconds with probability $(1-p)$ $(0.1 < p < 0.9)$. Unless otherwise noted, we assume $p = 0.3$ and the number of nodes is 30. The speed of the nodes follows a uniform distribution between 1.0 m/s and 2.0 m/s assuming walking speed of pedestrians. Omni-directional signal transmissions are assumed for both TDoA measurement and wireless communication. We set the maximum range R of RTM messages to $2r$, where r is the maximum range of TDoA measurement signals. By default, we set $r = 6$m. The range measurement error has a zero-mean Gaussian noise with standard deviation of σ. σ is defined as $\sigma = kd$ where d represents the distance between the nodes. The default value of the coefficient k is 0.01. Also, σ_0 in Eq. (3.2) is set to the same value as k. We set the maximum backoff time on sending RTM messages (CW) to 10ms, the waiting time for *moving* nodes to send reply messages to 10ms, and the cycle time to perform a localization to 70ms. The parameters for deriving initial solutions are configured as $\delta_{max} = 100/\text{m}^2$, $\delta_{min} = 10/\text{m}^2$ and $M_{max} = 15$. The other parameter settings are listed in Table 3.1.

With the settings above, we have conducted simulations of 3,000 seconds, and evaluated two metrics: *localization errors* and *tracking errors*. The localization error is defined by the average distance of true positions and estimated positions at the time when localization is performed. The tracking error is defined by the average position error of all the nodes which is evaluated *every second* through the whole simulation. We compared the

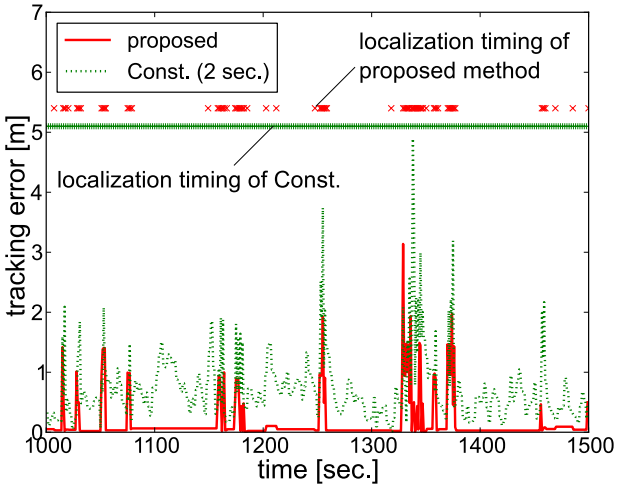


Figure 3.5: Time variation of tracking errors

performance of the proposed method with that of conventional cooperative localization methods (called *Const.*) in which nodes perform localization at constant time intervals of 2, 4, or 6 seconds, using all the neighboring nodes as pseudo-anchors regardless of their movement state.

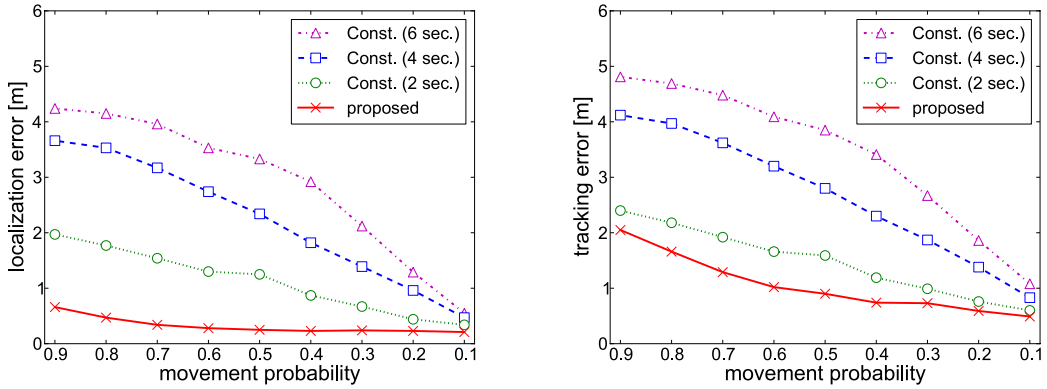
3.3.2 Simulation Results

Results Overview

Since position error of a node is accumulated as it moves, tracking performance depends not only on accuracy of each position estimation but also on the timing of position updates. Fig. 3.5 shows time variation in the tracking errors of a node when the movement probability of each mobile node is $p = 0.3$. The timing of position updates is also plotted above. For *Const.*, the tracking error continues being at a high level since the large position error of moving nodes propagates to the others. In contrast, the proposed method keeps high localization accuracy by the pseudo-anchor selection mechanism, so that position error of each node approaches zero while the node is stopping. Also, the plot of localization timing shows that the estimated position is updated only when the node is moving, and thus the total number of localization attempts is effectively reduced without degrading the tracking performance. In the following sections, we evaluate the performance of the proposed method from various aspects to show its effectiveness.

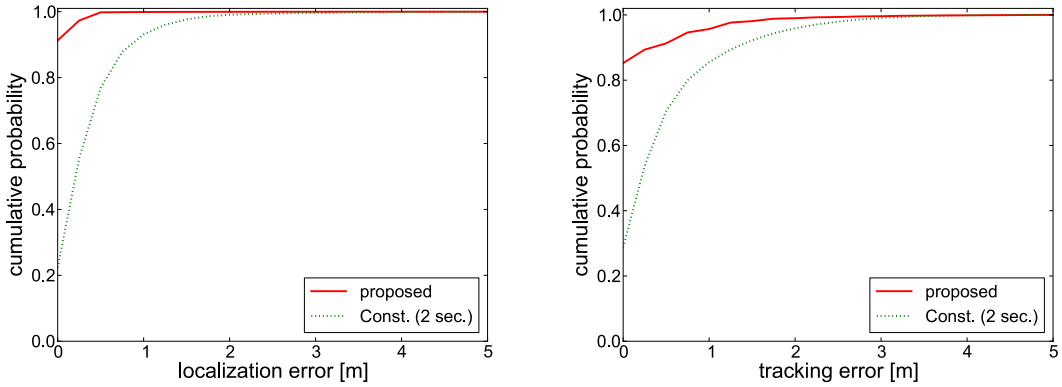
Localization Performance

First, we evaluated localization performance of the proposed method. Fig. 3.6 shows average localization errors and tracking errors as the movement probability is varied between 0.1 and 0.9. It can be seen that both errors of the compared methods are larger than



(a) average localization errors (b) average tracking errors

Figure 3.6: Tracking performance with different movement probabilities



(a) localization errors (b) tracking errors

Figure 3.7: Error distributions

those of the proposed method. This is because the compared methods use moving nodes as pseudo-anchors, which causes propagation of their large position errors. On the other hand, the proposed method could achieve sufficient accuracy by selecting only *static* nodes as pseudo-anchors. Fig. 3.7 shows the cumulative distributions of the localization errors and tracking errors when the movement probability is 0.3. As seen, the tracking error of the proposed algorithm is less than 1m for more than 95% of the experiment time, whereas the corresponding ratio with *Const.* (2 sec.) is less than 80%.

Movement Detection Capability

To demonstrate efficacy of the state validation algorithm, we also evaluated *movement detection rate* of the mobile nodes. Here, we consider the situations where some neighbors in *static* state have actually moved since their last localization round. In that case, they reply a measurement report message in response to the range measurement signals and become inappropriate pseudo-anchor candidates. The movement detection probability is defined as the ratio between the number of neighbors that are successfully detected as

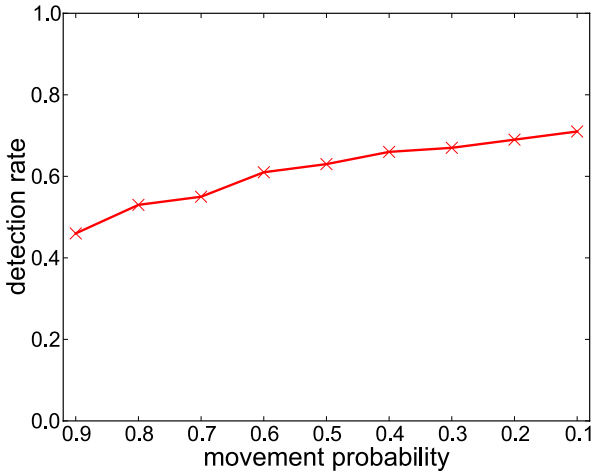


Figure 3.8: Success rate of movement detection

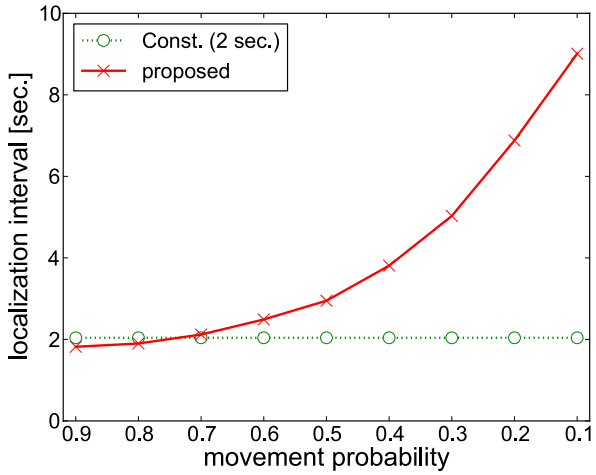


Figure 3.9: Average localization intervals

moving in the state validation process and the total number of such inappropriate pseudo-anchor candidates. Fig. 3.8 shows the movement detection probability with different movement probabilities. We can see that the detection probability is no less than 0.46 even if all the nodes move almost continuously (*i.e.*, with a movement probability of $p = 0.9$). The movement detection probability almost linearly increases as the movement probability decreases, since the inappropriate pseudo-anchor candidates can be detected more robustly as more *static* neighbors are actually stopping at their current estimated positions.

Localization Intervals

Frequency of localization attempts is also an important performance metric, because it directly affects the energy consumption on the mobile devices. Fig. 3.9 shows the average

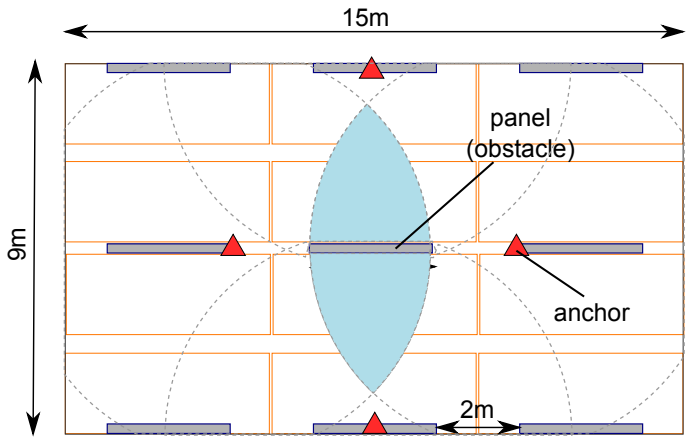


Figure 3.10: Field map (poster session scenario)

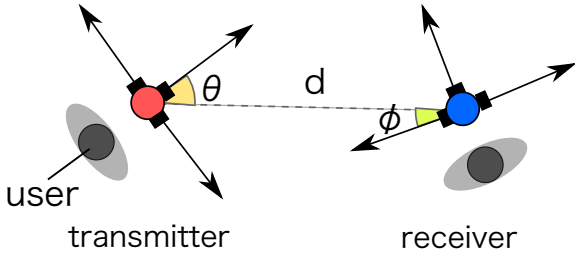
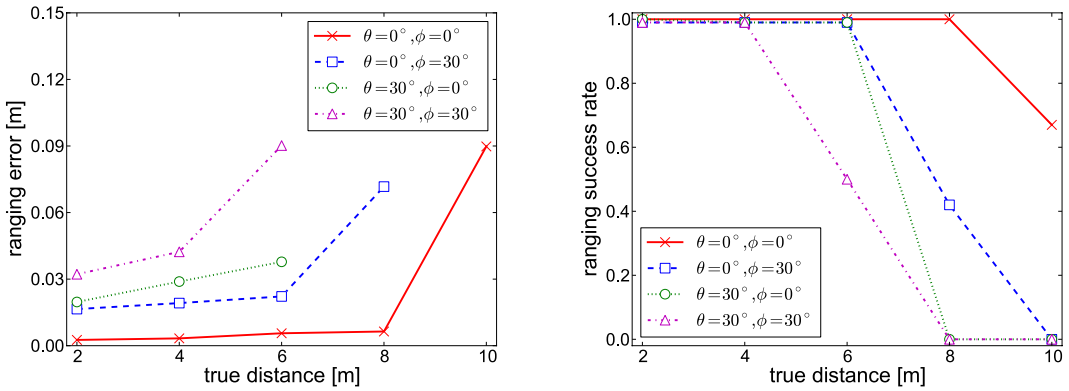


Figure 3.11: Node model

localization intervals with each movement probability. Since *Const.* performs localization at regular time intervals, its localization frequency is constant regardless of the movement probability. In contrast, the proposed method could reduce the localization frequency more effectively as the movement probability becomes lower, since it suppresses position updates of the nodes while they are stopping at the current location. In the proposed method, the total number of localization attempts could be reduced by up to 77% without degrading the tracking performance.

3.4 Performance in a Practical Scenario

In this section, we demonstrate that the proposed method works well under a real application scenario. We assume that a conference poster session with 12 poster panels held in a 9m × 15m hall as shown in Fig. 3.10. Each node is equipped with three pairs of ultrasound transmitters and receivers. We also assume that poster panels and human bodies obstruct propagation of range measurement signals. We model a human body as a 30cm line that is 30 cm away from the mobile device as shown in Fig. 3.11. Thus nodes can measure the distance to the neighbors in all directions except for their backward.



(a) average ranging error (b) ranging success rate

Figure 3.12: Range measurement results with MCS410CA

3.4.1 Modeling Range Measurement Errors

To evaluate performance of the proposed method in realistic environment, we should build a realistic model of ultrasound sensors on mobile nodes. We represent the model by the range measurement error and the ranging success rate. On account of high directionality of ultrasound signals, they depend on angle of departure (θ) and angle of arrival (ϕ) in Fig. 3.11, in addition to the distance d between the nodes. Hence, we assume that the range measurement error follows a Gaussian distribution whose mean and variance are defined as functions $\mu(d, \theta, \phi)$ and $\sigma(d, \theta, \phi)$, respectively, while the ranging success rate is determined by a function $\rho(d, \theta, \phi)$. We define these functions based on range measurement experiments using Cricket MOTE (MCS410CA) [14].

At first, we evaluated the range measurement error and ranging success rate between two static nodes for each combination of d, θ, and ϕ. We varied d from 2m to 10m in increments of 2m, and also varied θ and ϕ from 0° to 45° in increments of 15°. The measurement was conducted 300 times for each combination. We show a part of measurement results in Fig. 3.12. From the results, we can see that directionality of ultrasound signals has large impact on range measurement accuracy.

To construct a realistic sensor model, we should also consider slight movement of the sensors due to walking motion, which also may affect the range measurement. Therefore, we evaluated the ranging success rate when a mobile node is moving at a speed of 1 m/s toward a static node transmitting ultrasound signals. In the experiment, a person holding a Cricket MOTE walked toward a static node placed at the same height, as shown in Fig. 3.13. We varied ϕ from 0° to 45° in increments of 15° and measured 10 times for each ϕ. As a result, we confirmed that in all the cases of ϕ, the maximum transmission range of measurement signals were about 2m shorter than those in the static case (see Fig. 3.14). Hence, for moving nodes, we calculated the mean and variance of range measurement errors and ranging success rate by assigning $(d+2.0, \theta, \phi)$ instead of (d, θ, ϕ) to the ranging model for static nodes.

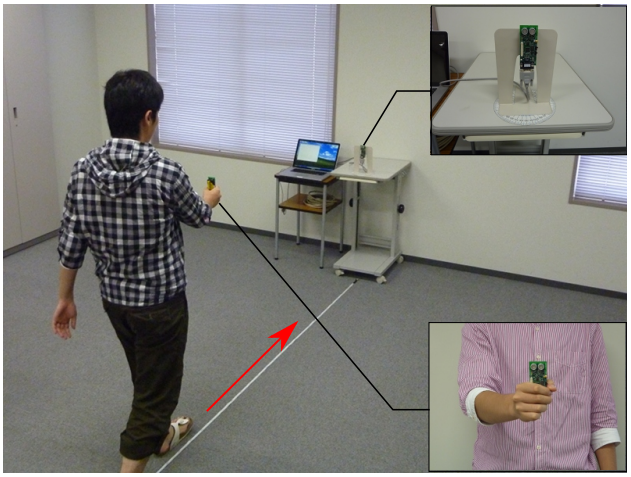


Figure 3.13: Ranging experiment (walking)

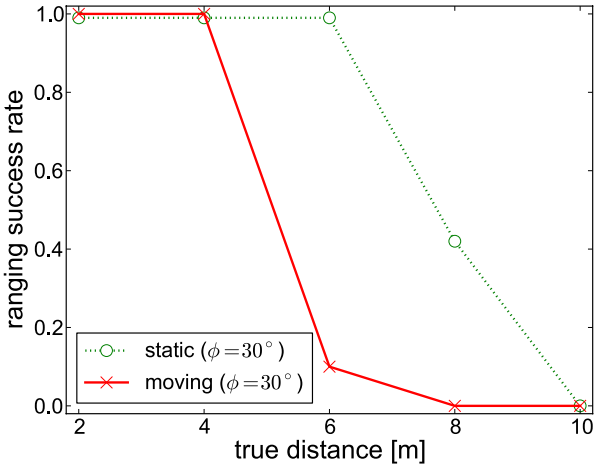


Figure 3.14: Degradation of ranging success rate due to walking motion

3.4.2 Obtaining Mobility Traces

To obtain actual mobility traces of presenters and audiences, we also conducted field experiments as shown in Fig. 3.15. 12 students behaved as audiences and presenters in a 9m $\times$ 15m hall where 12 poster panels were arranged as Fig. 3.10. We laid out 1,500 markers that are made of 0.3m-square papers printed with their coordinate on the ground. Each student moved over the markers recording the coordinate with a video camera for 5 minutes. After conducting such experiments 8 times, we obtained 72 traces of audiences and 24 trajectories of presenters.

3.4.3 System Performance

We conducted simulation experiments using the sensor model in Section 3.4.1 and actual mobility traces in Section 3.4.2. 4 anchors are deployed at (4.0, 4.5), (7.5, 0.0), (11.0,

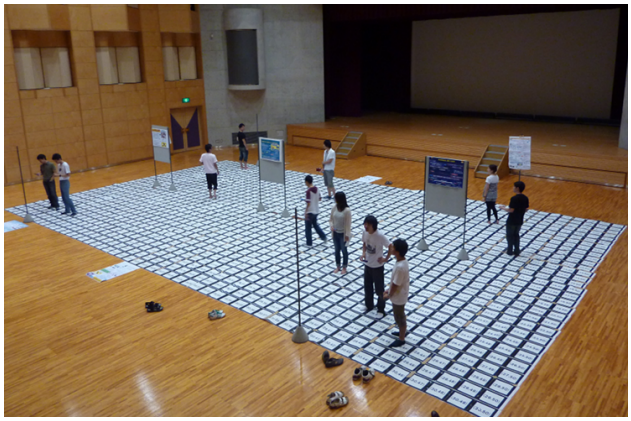


Figure 3.15: Field experiment

Table 3.2: Simulation results (poster session scenario)

	Proposed	**Const.**	**Recursive**
Estimation Error	0.23m	0.85m	4.68m
Tracking Error	0.51m	0.95m	5.03m
Localization Success Rate	0.95	0.99	1.00
Avg. Localization Interval	5.49 sec.	2.00 sec.	2.35 sec.
Context Recognition Rate	0.89	0.74	0.46

4.5) and (7.5, 9.0) as shown in Fig. 3.10. The shaded region indicates the area where the nodes can find three or more anchors in case that the maximum range of measurement signals is 5m. We assumed that fixed anchors are deployed on the wall or poster panels, where ranging signals are not obstructed by human bodies. We selected 12 trajectories of presenters to place them for each poster and 38 trajectories of audiences to conduct a simulation for 5 minutes. In Table 3.2, we compare the performance of the proposed method with two conventional cooperative approaches. In the first method (referred to as *Const.*), each node updates its location every 2 seconds using all the neighboring nodes as pseudo-anchors. The second one (referred to as *Recursive*) is based on DOLPHIN [29], in which each anchor/pseudo-anchor transmits ranging signals in a random order while the other nodes immediately perform localization and serve as pseudo-anchors after they collect distance information from a sufficient number of reference points. Since DOLPHIN is a cooperative method for static sensor networks, we deleted the estimated positions of the nodes every 2 seconds and repeatedly performed localization to track the mobile nodes.

For *Recursive*, errors are remarkably large since moved pseudo-anchors seriously degrade localization accuracy. As for *Const.*, the deterioration caused by bad pseudo-anchors happens to be relaxed by averaging observations from all the neighboring nodes, whereas relatively large errors still remain. In contrast, the proposed method improved localization accuracy by 73% compared to *Const.* and by 95% to *Recursive* as a benefit of pseudo-anchor selection based on movement state estimation. Furthermore, the total number

of localization attempts is reduced by 64% and 57%, respectively, which proves that the proposed method achieves high localization efficiency.

Such accuracy enhancement contributes to not only pedestrian navigation but also various location-based applications. For example, here we consider a context-aware service that identifies the poster in which the user is interested. The poster of interest is estimated based on user's position; if the estimated position is in the 2m×5m-sized rectangular region in front of each poster (the regions are also indicated in Fig. 3.10), the user is considered to be seeing it. Thus for each node we estimated the poster of interest every second to evaluate accuracy rate of the context recognition. The results in Table 3.2 demonstrate that the context recognition rate is directly affected by the position accuracy. The proposed method successfully enhanced the recognition capability by 15% to *Const.*, and 43% to *Recursive*.

3.5 Conclusion

This chapter has presented a fully-decentralized algorithm to collaboratively localize mobile nodes with a small number of anchor nodes and a reduced amount of localization attempts. Focusing on the stop-and-go behavior of mobile nodes, which is typically followed by people attending exhibitions, the proposed method detects movement of the nodes and selects only *static* nodes as pseudo-anchors. Thus it effectively prevents error propagation from the nodes in motion. Furthermore, it automatically adjusts localization frequency according to the estimated speed of each mobile node to reduce redundant localization attempts. Experimental results have shown that the average localization error of the proposed method is around 0.2m, which is substantially lower than that of existing cooperative localization methods. We have also shown that the method could reduce localization frequency by up to 77% without degrading the tracking performance. Efficacy in a practical scenario has been also confirmed through experiments using a realistic sensor model and real human traces.

Although we have assumed ultrasound-based distance measurement in design and evaluation of the proposed localization system, its basic idea is essentially not dependent on any specific ranging techniques. Other measurement means could be also applied by slightly modifying the protocol for neighbor collaboration. For pedestrian tracking, it would be desirable that localization can be completed using only built-in functions of commercial mobile devices (*e.g.*, smartphones). Combination with audio-based ranging techniques [70, 71] would be a promising approach in this direction, if their calibration effort for device-dependent tuning could be mitigated somehow. In addition, the idea of utilizing static neighbors as pseudo-anchors would be effective not only for pedestrian tracking but also for a wider range of applications where target objects move in a stop-and-go manner. A typical example would be package tracking in a distribution warehouse; since most of the packages are expected to be kept at designated locations, they are usually stationary for a long period of time and thus serve as reliable pseudo-anchors. If dedicated wireless tags are available for such object tracking, implementation based on UWB radio would

also be a reasonable option since it enables accurate distance measurement with a longer signal transmission range than ultrasound or audio-based systems. These extensions would further enhance applicability of the proposed localization algorithm to practical scenarios.

Chapter 4

Performance Analysis and Improvement of Mobility-Aware Cooperative Localization

4.1 Introduction

The basic idea of mobility-aware cooperative localization is to incorporate observations on common characteristics of human mobility into the algorithm design of localization systems to break the trade-off between accuracy and their costs (*i.e.*, infrastructure deployment and calibration effort). As an instance of the concept, we have focused on the stop-and-go behavior of indoor pedestrians and designed an energy-efficient cooperative localization algorithm in Chapter 3.

In this chapter, we design a general framework for performance analysis of the mobility-aware cooperative localization algorithms that exclude the nodes in motion from the set of reference points. In order to clarify when and how they can achieve optimal performance, we derive the theoretical lower bound of the localization errors, assuming that (i) nodes in motion can be excluded from the set of pseudo-anchors, (ii) position errors of the pseudo-anchors follow independent Gaussian distributions and (iii) measured distance between the nodes has a zero-mean Gaussian noise. Through a case study, we compare performance of the proposed localization algorithm in Chapter 3 with the theoretical bound, and show that the errors asymptotically approach the lower bound as more than 50% of neighboring devices maintain correct movement state. In addition, we further extend the proposed localization algorithm based on the observations from the theoretical analysis, in order to achieve better positioning accuracy. The framework of error analysis is also applicable to any other cooperative localization algorithms that select pseudo-anchors based on their movement state.

As well as the error analysis, performance of the mobility-aware approaches is also analyzed from various aspects, seeking strategies to maximize their effectiveness. Based on extensive simulations and experiments using Android smartphones, we show several important observations regarding anchor deployment strategy, effectiveness of accelerometer-

based motion detection and combination with PDR-based trajectory estimation, etc.

The rest of this chapter is organized as follows. Section 4.2 presents the theoretical analysis of the mobility-aware approach based on the movement state detection, followed by a case study. In Section 4.3, we extend the proposed algorithm in Chapter 3 based on observations from the error analysis. Section 4.4 shows the results of our extensive simulations and experiments using Android smartphones. Finally, Section 4.5 concludes this chapter.

4.2 Error Analysis

4.2.1 Deriving Theoretical Error Bound

Given the ranging error model and position error distribution of pseudo-anchors, we can derive the lower bound of localization errors in each localization attempt. For the analysis in this section, we assume the following three conditions:

(i) Nodes in motion can be excluded from the set of pseudo-anchors.

(ii) Position errors of the pseudo-anchors follow independent Gaussian distributions.

(iii) Measured distance between the nodes has a zero-mean Gaussian noise.

In locating a node, say A_0, its neighbors in *static* state serve as pseudo-anchor candidates while some of them may have wrong position due to undetected movements. Assume that we have m pseudo-anchor candidates where m_s of them are actually stopping at their estimated positions, whereas the remaining $(m - m_s)$ candidates have actually moved and their movement has yet to be detected. By the pseudo-anchor selection mechanism (*e.g.*, the one we have designed in Section 3.2.3), we can eliminate such bad pseudo-anchor candidates and thus, ideally, the m_s *genuine* static neighbors are to be selected as pseudo-anchors.

Here we assume such ideal pseudo-anchor selection, and define the location parameter vectors as follows:

$$\boldsymbol{\alpha}_0 = [x_0, y_0] \tag{4.1}$$

$$\boldsymbol{\alpha}_A = [x_1, y_1, x_2, y_2, \ldots, x_{m_s}, y_{m_s}] \tag{4.2}$$

$$\boldsymbol{\alpha} = [\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_A]^T \tag{4.3}$$

where $\boldsymbol{\alpha}_0$ is the position of A_0, and $\boldsymbol{\alpha}_A$ contains current estimated positions of the selected pseudo-anchors. Since the positions of the pseudo-anchors in $\boldsymbol{\alpha}_A$ have some uncertainty, we put them into the parameter vector $\boldsymbol{\alpha}$ as well as $\boldsymbol{\alpha}_0$.

Let $\widehat{\boldsymbol{\alpha}}$ be an estimate of parameter vector $\boldsymbol{\alpha}$. Then the error covariance matrix $\boldsymbol{C}$ is defined as

$$\boldsymbol{C} = E\{(\widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha})(\widehat{\boldsymbol{\alpha}} - \boldsymbol{\alpha})^T\}. \tag{4.4}$$

This error covariance matrix is bounded below as [88]:

$$\boldsymbol{C} \geq \boldsymbol{J}_T^{-1} \tag{4.5}$$

where $\boldsymbol{J}_T$ is the Fisher Information Matrix provided by measurement vector $\boldsymbol{X}$ and the position error distribution of the pseudo-anchors.

Let $\widehat{d}_k$ be the measured distance to the kth pseudo-anchor. The measurement vector $\boldsymbol{X}$ is defined as

$$\boldsymbol{X} = \left[\widehat{d}_1, \widehat{d}_2, \ldots, \widehat{d}_{m_s}\right]. \tag{4.6}$$

Assuming that measurement noise is Gaussian, the probability density function (PDF) of the measurement vector $\boldsymbol{X}$ is also Gaussian and given by

$$f_X(\boldsymbol{X}; \boldsymbol{\alpha}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\alpha}), \boldsymbol{\Sigma}) \tag{4.7}$$

where $\boldsymbol{\mu}(\boldsymbol{\alpha})$ is a vector of true distances. In the simulations below, we assume that the measurements are not correlated, and thus the covariance matrix $\boldsymbol{\Sigma}$ is diagonal. The diagonal elements of $\boldsymbol{\Sigma}$ are defined as $\boldsymbol{\Sigma}_{k,k} = (\sigma_0 d_k)^2$, where d_k is the true distance to the kth pseudo-anchor A_k, and σ_0 is the standard deviation of distance measurement errors with $d_k = 1.0$ m.

For the given PDF $f_X(\boldsymbol{X}; \boldsymbol{\alpha})$, the information matrix provided by measurement vector $\boldsymbol{X}$, say $\boldsymbol{J}(\boldsymbol{\alpha})$, is defined as follows:

$$\boldsymbol{J}(\boldsymbol{\alpha}) = E\left\{\left[\nabla_{\boldsymbol{\alpha}} \ln f_X(\boldsymbol{X}; \boldsymbol{\alpha})\right]\left[\nabla_{\boldsymbol{\alpha}} \ln f_X(\boldsymbol{X}; \boldsymbol{\alpha})\right]^T\right\}. \tag{4.8}$$

We also model the position error distribution of pseudo-anchors by Gaussian:

$$f_A(\boldsymbol{\alpha}_A) = \mathcal{N}(\boldsymbol{\mu}_A, \boldsymbol{\Sigma}_A) \tag{4.9}$$

where $\boldsymbol{\mu}_A$ and $\boldsymbol{\Sigma}_A$ indicate the vector of pseudo-anchor positions and their uncertainty, respectively.

The PDF $f_A(\boldsymbol{\alpha}_A)$ provides prior information about the parameter vector $\boldsymbol{\alpha}_A$ as

$$\begin{aligned} \boldsymbol{J}_A &= E\left\{\left[\nabla_{\boldsymbol{\alpha}} \ln f_A(\boldsymbol{\alpha}_A)\right]\left[\nabla_{\boldsymbol{\alpha}} \ln f_A(\boldsymbol{\alpha}_A)\right]^T\right\} \\ &= \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & {\boldsymbol{\Sigma}_A}^{-1} \end{bmatrix}. \end{aligned} \tag{4.10}$$

Since information provided by the measurements and a priori information are assumed to be independent, they can be summed as [88]:

$$\boldsymbol{J}_T = \boldsymbol{J}(\boldsymbol{\alpha}) + \boldsymbol{J}_A. \tag{4.11}$$

Based on Eq. (4.5), the covariance lower bound of A_0's estimated position can be computed as

$$\boldsymbol{\Sigma}_{p_0} = \begin{bmatrix} {\boldsymbol{J}_T^{-1}}_{1,1} & {\boldsymbol{J}_T^{-1}}_{1,2} \\ {\boldsymbol{J}_T^{-1}}_{2,1} & {\boldsymbol{J}_T^{-1}}_{2,2} \end{bmatrix}. \tag{4.12}$$

Thus the localization error bound e_{p_0} of A_0 can be computed as:

$$e_{p_0}^2 = tr(\boldsymbol{\Sigma}_{p_0}). \tag{4.13}$$

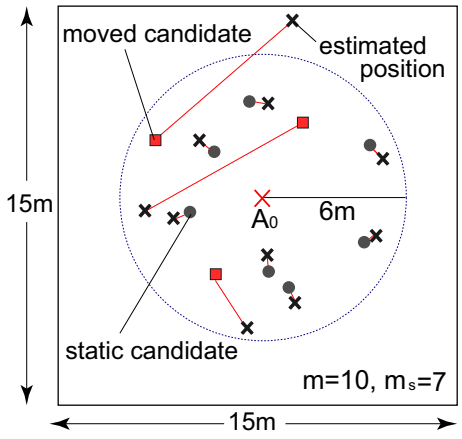


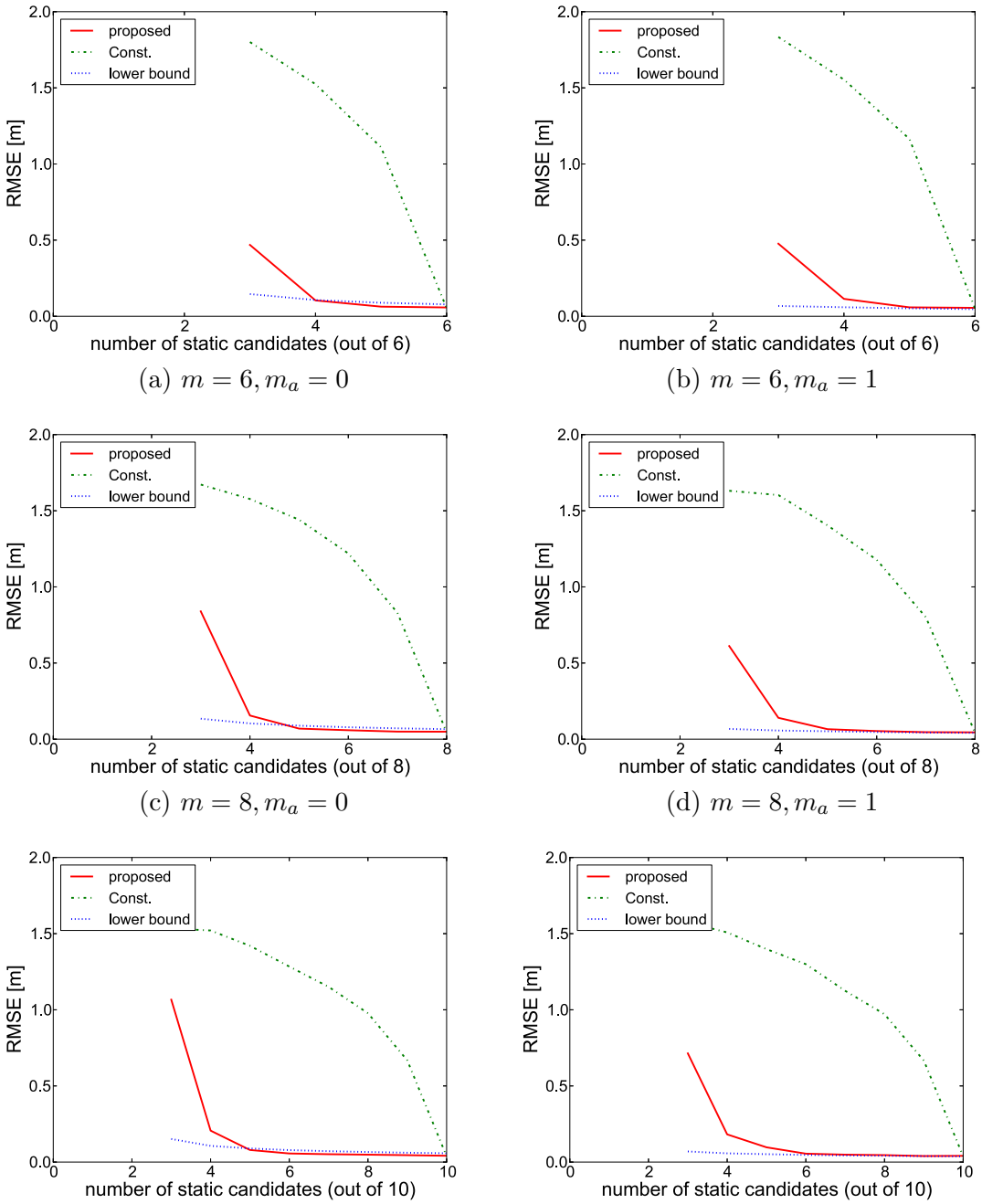
Figure 4.1: Simulation settings (error analysis)

4.2.2 Case Study

Through simulations with various node topologies, we compared the localization error of the proposed method to the theoretical bound in Eq. (4.13). Here we consider a localization process of a node A_0, which is currently at the center of a 15m×15m-sized field as shown in Fig. 4.1. Assume that A_0 has received current positions and measured distances from m pseudo-anchor candidates within its radio range (6m), where m_s of them are actually stopping at their estimated positions. We assume that the estimated positions of these m_s static nodes follow Gaussian distributions centered at their true positions whose covariance matrices are given by $\overline{\sigma_p}^2 \boldsymbol{I}_{2\times 2}$, where $\boldsymbol{I}_{2\times 2}$ is a two-dimensional identity matrix. $\overline{\sigma_p}$ is a parameter to characterize position uncertainty of the pseudo-anchors, and we set $\overline{\sigma_p} = 0.08$ based on the results in section 3.3.2. The remaining $(m - m_s)$ candidates have actually moved, so that we randomly chose their estimated positions from the whole field.

For various combinations of m and m_s, we randomly generated 1,000 topologies of pseudo-anchor candidates. Then for each instance, we estimated A_0's position by the proposed method and *Const.* to compare the localization error with the lower bound derived by Eq. (4.13). Fig. 4.2 (a), (c) and (e) show the average localization errors of these algorithms and the corresponding theoretical bounds with $m = 6$, 8 and 10, respectively. For *Const.*, the estimation accuracy is drastically degraded even if only one of m pseudo-anchor candidates has moved (*i.e.*, $m_s = m - 1$). In contrast, the localization error of the proposed method approaches the theoretical lower bound when more than 50% of pseudo-anchor candidates are actually stopping at their estimated positions. On the other hand, the localization error steeply rises beyond the lower bound when the ratio of the static candidates drops below this boundary, which may be mainly due to the failure of pseudo-anchor selection. Similar results were also observed for other settings of m, so we concluded that the proposed algorithm requires at least 50% of pseudo-anchor candidates to be stopping at their estimated positions to select appropriate pseudo-anchors.

When the set of pseudo-anchor candidates contains some fixed anchors, we can nar-



(a) $m = 6, m_a = 0$ (b) $m = 6, m_a = 1$

(c) $m = 8, m_a = 0$ (d) $m = 8, m_a = 1$

(e) $m = 10, m_a = 0$ (f) $m = 10, m_a = 1$

Figure 4.2: Average localization error as a function of the number of static pseudo-anchor candidates

row down the solution space of the estimated position to the circular rings for the fixed anchors instead of considering all the neighbors in *static* state, since the anchors are definitely stopping at their true positions. It lessens the possibility that a wrong set of *static* neighbors is selected as pseudo-anchors and thus contributes to enhance the robustness of pseudo-anchor selection. Fig. 4.2 (b), (d) and (f) show the localization error where the set of pseudo-anchor candidates contains one fixed anchor ($m_a = 1$), where other settings correspond with those of (a), (c) and (e), respectively. In each case, we can see that accuracy degradation around low m_s, which would be mainly due to the failure of pseudo-anchor selection, is mitigated. This implies that if the fixed anchors are deployed so that the nodes can measure the distance from at least one of them, the position accuracy would be enhanced effectively.

4.3 Improving State Validation Algorithm

In section 4.2.1, we have derived the error bound when the nodes in motion are excluded from the reference points for localization. While the position errors may be larger than the theoretical bound, we have also shown in section 4.2.2 that it exhibits strong correlation with the actual errors as far as a sufficient proportion of pseudo-anchor candidates (*i.e.*, the neighboring nodes that are currently in *static* state) are actually stopping at their estimated positions. Note that this condition would hold in many cases since movement of the nodes is collaboratively monitored among the neighboring nodes and thus their movement state is usually updated within a sufficiently short delay. Thus the theoretical bound would serve as a reasonable error indicator in practical scenarios. Therefore we have decided to replace σ_{p_j} in Eq. (3.1), which is the expected position error of each pseudo-anchor candidate A_j, by e_{p_0} in Eq. (4.13):

$$\sigma_{p_j} = e_{p_j} = tr(\mathbf{\Sigma}_{p_j}) \tag{4.14}$$

where $\mathbf{\Sigma}_{p_j}$ is given by Eq. (4.12). In order to derive the covariance lower bound $\boldsymbol{J}_T$, we need (i) estimated positions $\boldsymbol{\phi}_j$ and their covariance matrices $\mathbf{\Sigma}_{p_j}$ of the pseudo-anchor candidates A_j and (ii) true distance d_j between the node of interest A_i and each candidate A_j. The covariance matrix of each pseudo-anchor A_j, say $\mathbf{\Sigma}_{p_j}$, is reported to A_i with $\widehat{d}_j$ and $\boldsymbol{\phi}_j$ as a response to its measurement signals. Then we diagonally arrange each two-dimensional covariance matrix $\mathbf{\Sigma}_{p_j}$ to form a $2m_s \times 2m_s$ matrix $\mathbf{\Sigma}_A$ (see Eq. (4.9)). Here we assume that fixed anchors have substantially small uncertainty of $10^{-5}\boldsymbol{I}_{2\times 2}$, where $\boldsymbol{I}_{2\times 2}$ is a two-dimensional identity matrix. Since d_j is unknown in the localization process, here we replace d_j by the measured distance $\widehat{d}_j$ in the uncertainty estimation in Eq. (4.7).

Based on the improved uncertainty estimation mechanism, we determine the threshold ϵ_j by Eq. (4.15), instead of Eq. (3.1):

$$\epsilon_j = \sqrt{(\alpha\sigma_{r_j})^2 + (\beta\sigma_{p_j})^2}. \tag{4.15}$$

The coefficients α and β are parameters to determine sensitivity of the movement detection.

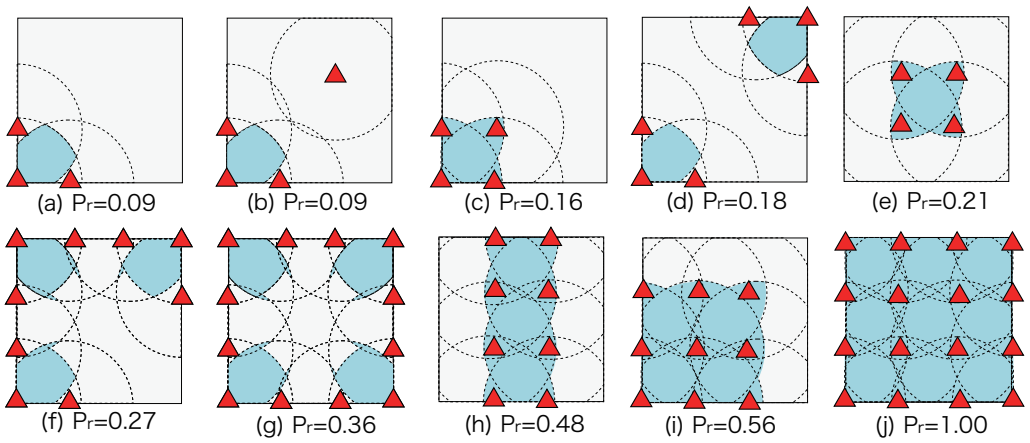


Figure 4.3: Anchor deployment patterns

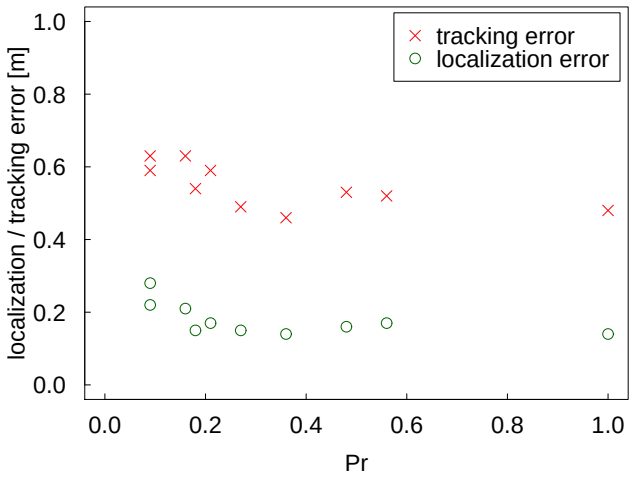


Figure 4.4: Impact of anchor deployment patterns

Unless otherwise noted, we set $\alpha = 3.0$ and $\beta = 3.0$ in the evaluations in the following sections.

4.4 Performance Analysis

In this section, we analyze performance of the extended algorithm from various aspects including energy efficiency, computational cost and robustness against node density to show its effectiveness in practical environments. Unless otherwise noted, we employ the same simulation settings as in Section 3.3.1. Utilization of inertial sensors in mobile devices and combination with PDR technology are also considered to enhance its applicability to various situations in Sections 4.4.8 and 4.4.9, respectively.

4.4.1 Anchor Deployment Pattern

Since fixed anchors provide original reference points in our cooperative localization system, spatial patterns and coverage of the anchors would substantially affect the positioning

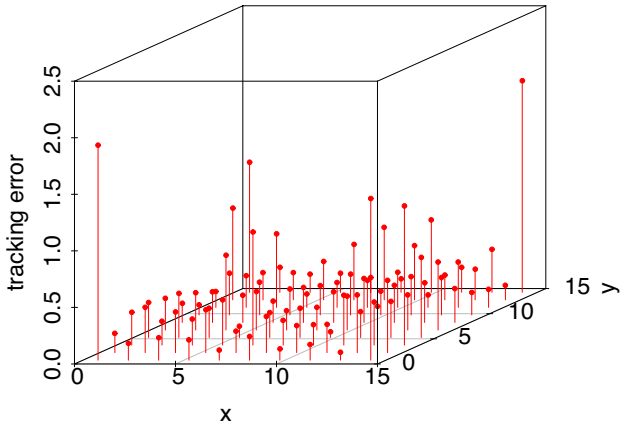


Figure 4.5: Spatial distribution of tracking errors

performance. To examine the impact of the anchor deployment patterns, we conducted simulations with 10 different scenarios in Fig. 4.3. The shaded region in the figure indicates the area where nodes can find three or more anchors, which means that they can be localized without relying on pseudo-anchors. To quantify difficulty of accurate position estimation with each deployment pattern, we introduce an index P_r which is defined by average ratio of time when the nodes stay in the shaded area during the whole simulation time.

Fig. 4.4 shows average localization errors and tracking errors of our method with each deployment pattern. Both localization errors and tracking errors tend to decrease as P_r is low. The proposed method could achieve the localization error of 0.28m and the tracking error of 0.63m on average even in case of $P_r = 0.09$. Furthermore, in the case with $P_r = 0.27$, we could achieve almost equivalent accuracy as $P_r = 1.00$.

We also evaluated spatial distribution of tracking errors with the pattern (e) by separately averaging tracking errors for every 1.5m $\times$ 1.5m area. In Fig. 4.5, we can see that tracking errors of the nodes near the boundary of the field were relatively larger because it would often happen that the nodes could not obtain a sufficient number of accurate reference points.

4.4.2 Node Density

To examine the impact of node density on the tracking performance, we evaluated tracking errors, varying the number of nodes N from 10 (0.04 nodes/m^2) to 100 (0.4 nodes/m^2). The solid line in Fig. 4.6 shows the average tracking error in each case. When the number of nodes is no less than 20, the tracking error converges to a certain level (about 0.6m in this scenario). On the other hand, if the number of nodes is less than 20, the tracking error steeply rises since the success rate of localization starts declining. As seen in the previous section, such failure of localization is more likely to occur near the boundaries of the field, since nodes can find relatively fewer neighbors within their transmission range of the measurement signals. To guarantee the tracking performance in such sparse environment,

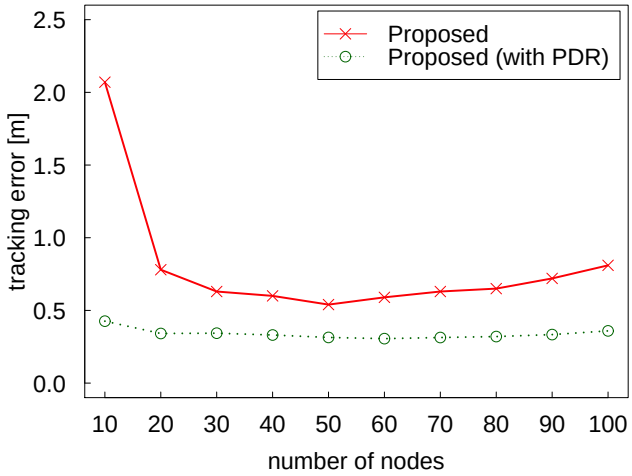


Figure 4.6: Impact of node density

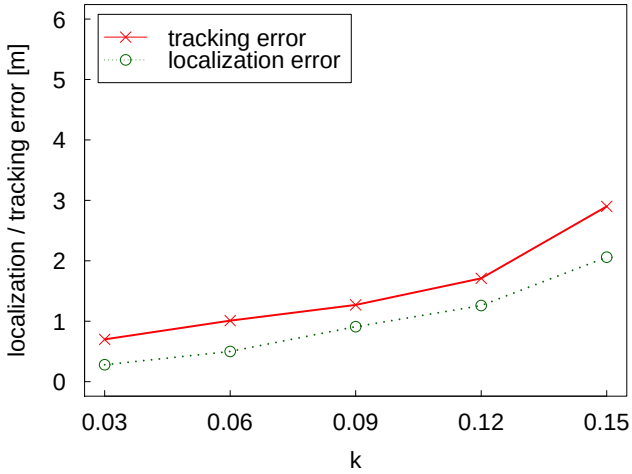


Figure 4.7: Impact of distance measurement error

we need to arrange more anchors so that the nodes can find a sufficient number of reference points anywhere in the field. Thus the number of anchors that are required to achieve robust tracking depends on both density and mobility of mobile nodes.

In Fig. 4.6, we can also see that the tracking error gradually rises when N is larger than 60. This is because the higher node density extends the latency for localization since many nodes simultaneously attempt localization. However, the negative effect of such latency on the tracking performance happens to be limited under reasonable density, considering that the tracking error with $N = 100$ is only 0.22m larger than that with $N = 60$.

4.4.3 Range Measurement Error

To evaluate the impact of range measurement error, we conducted simulations varying k in Section 3.3.1 from 0.03 to 0.15. Note that the average ranging errors are within 1m for each case. Fig. 4.7 shows the localization error and the tracking error with each k. Although the proposed method can keep relatively high localization accuracy owing to

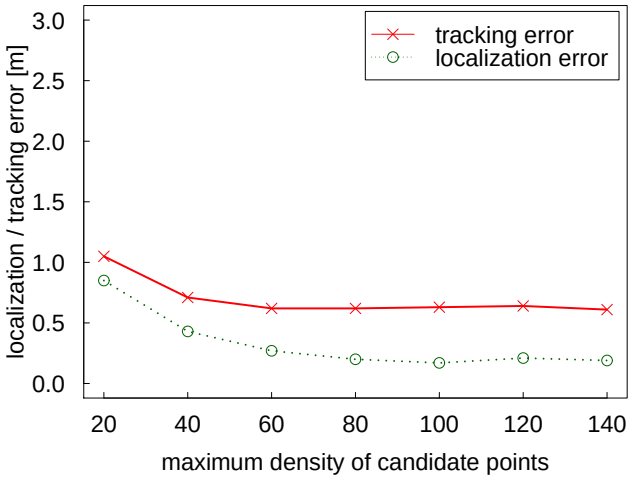


Figure 4.8: Impact of the density of candidate points

the pseudo-anchor selection mechanism, inaccurate range measurement makes it difficult to detect short movements of the nodes. Consequently, the errors gradually increase as the measurement error grows. Since allowable errors in common location-based services such as navigation would be at most a few meters, it would be desirable that ranging error does not exceed 1m in order to fully derive advantage from the collaborative movement detection.

4.4.4 Computational Cost

Computational cost is also an important feature for self-localization of mobile nodes due to their limited hardware resources. Toward better computational efficiency, we have examined the best trade off between the number of candidate points and robustness of the pseudo-anchor selection. Assuming that the total number of candidate points to find the initial solution is n and the number of pseudo-anchor candidates (or if any, the number of neighboring fixed anchors) is m, the computation complexity for the state validation process is given by $O(nm)$. Thus the computational cost primarily depends on the density of the candidate points. Too few points would lead to failure in the pseudo-anchor selection, while too many points may incur excessive computation time and resource consumption. Note that m cannot be controlled by the algorithm since it depends on density and mobility of the nodes. Fig. 4.8 shows localization and tracking errors of the proposed method when the maximum density of candidate points δ_{max} is varied from 20 to 140. Note that the average number of candidate points with $\delta_{max} = 20, 60$ and 100 was 246, 574 and 922, respectively. We can see that the localization performance converges at a certain level if δ_{max} is no less than 60. Otherwise the errors gradually increase due to the cases that no candidate points are generated in the intersection of the largest number of circular rings.

To clarify whether the proposed localization scheme can meet the real-time requirement with commercial mobile devices, we have implemented the proposed method on the Android platform and evaluated average computation time for each localization

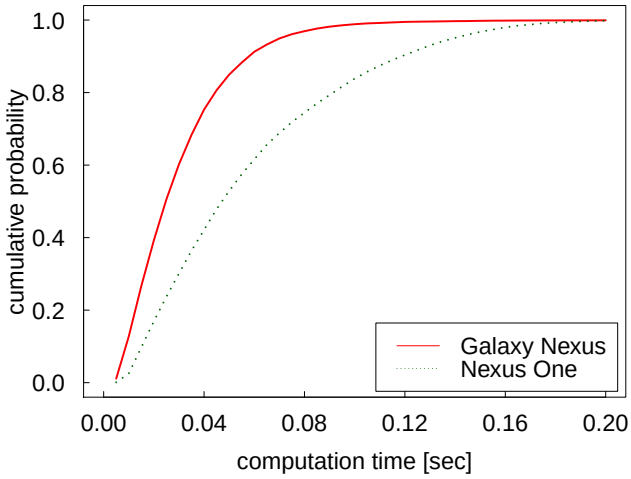


Figure 4.9: Computation time

Table 4.1: Energy model (MICAz [1])

	mode	**energy[mW]**
radio	send	52.2
	receive	59.1
	idle	0.06
	sleep	0.003
processor	active	24.0
	sleep	0.025

round (including both pseudo-anchor selection and position estimation). The program is written in the Java language and employs the Apache Commons Mathematics Library (`http://commons.apache.org/proper/commons-math/`) for matrix calculations and the least-squares optimization. For the experiment, we first ran a simulation with the default parameter setting and extracted the set of pseudo-anchor candidates and expected uncertainty of their estimated positions in all the completed localization rounds (We excluded the cases where a sufficient number of anchors are not found since localization is not performed in such cases). Accordingly, we have obtained more than 18,000 patterns of pseudo-anchor candidate sets and have applied the proposed method for each of them on two Android phones with different specifications; Samsung Galaxy Nexus (with Android OS 4.2.2, 1.2GHz dual-core CPU and 1GB RAM) and HTC Nexus One (with Android OS 2.3.6, 1.0GHz single-core CPU and 512MB RAM). Fig. 4.9 shows the cumulative distribution of the computation time with each phone. Average computation time was 0.031 seconds with Galaxy Nexus and 0.058 seconds with Nexus One. The results show that the proposed localization method can sufficiently meet the real-time requirement on off-the-shelf mobile devices.

4.4.5 Energy Consumption

Next, we identify the effectiveness of the proposed method in terms of energy consumption. In locating a node via cooperative localization, the main sources of battery consumption

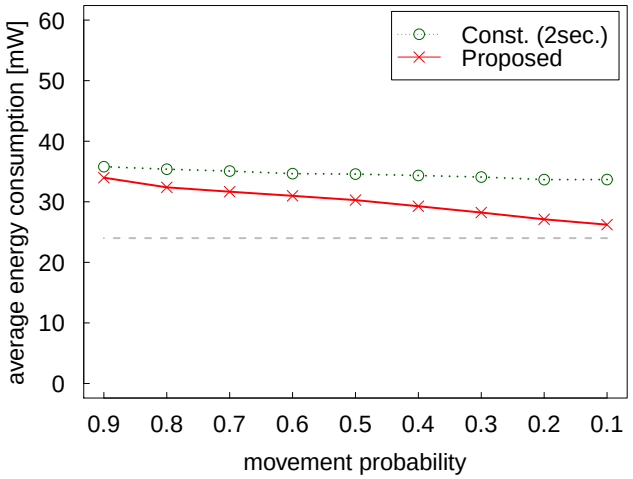
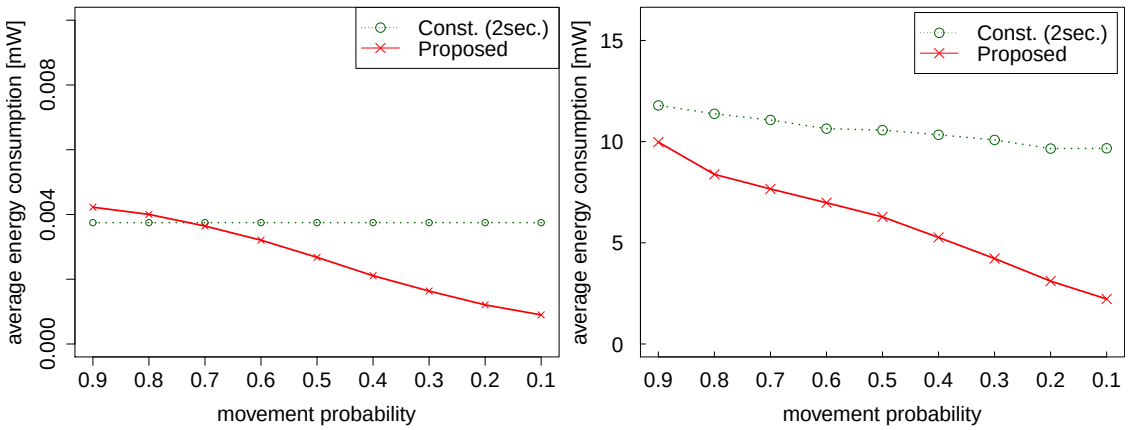


Figure 4.10: Total energy consumption

are i) wireless communications, ii) data processing, and iii) range measurement. Thus total energy consumption of mobile nodes depends on energy model of the corresponding hardware components. To save energy, modules in mobile devices usually support various power consumption modes. For example, radio module typically has four operation modes, namely, *transmit*, *receive*, *idle* and *sleep*. To approximate the energy consumption of the proposed method, we employ the energy model of MICAz MOTE [1], which is widely used for implementation of WSNs. It has an Atmega 128L processor as well as an IEEE 802.15.4 compliant RF transceiver to send and receive data at a rate of 250 kbps (the specific energy profiles are given in Table 4.1). Assuming that the power level is fixed for each operation mode, we can determine the total energy consumption based on active time of each energy profile. We assume that RTM, measurement report and movement notification messages have the same length of 32 bytes including headers. We also assume that the processor stays in *active* mode throughout the experiment, and radio module keeps *idle* mode to monitor the messages from neighbors while it is not sending or receiving data. For range sensing, we refer to the energy model of Cricket MOTE [14], in which ultrasound transducers are driven at 200mW to transmit ultrasound pulses with duration of 125μs.

Based on the assumptions above, we evaluated average energy consumption per second under the default simulation scenario in Section 3.3.1. Fig. 4.10 shows the total energy consumption by all the hardware components (processor, radio and ultrasound modules). The proposed method reduced the overall energy consumption by up to 22% compared to the *Const.* with a localization interval of 2 seconds. Since we assume that the processor stays in *active* mode throughout the simulation, data processing energy of the proposed method corresponds with that of *Const.* (the energy level is indicated by a dotted line in Fig. 4.10). Hence the 22% improvement in energy efficiency entirely comes from the optimization of localization frequency and the communication protocol.

Fig. 4.11 (a) represents the average battery consumption by ultrasound modules. Since the ranging energy increases in proportion to the number of localization attempts,



(a) ultrasound modules (b) wireless communications

Figure 4.11: Energy consumption by each component

energy consumption in the proposed method can be effectively reduced as the movement probability is low. However, the reduction of ranging energy would poorly contribute to the overall energy optimization because the energy level of ultrasound transducers is much smaller than that of other components.

Fig. 4.11 (b) shows the energy consumption by wireless communications. Although the proposed method requires additional message exchange for RTM and movement notification which are not necessary for *Const.*, it could totally reduce the power consumption by wireless communication by up to 77%. When the movement probability is low, the proposed method tends to assign longer localization intervals which contributes to reduce the message exchange. Although the localization frequency increases as the movement probability rises, the communication energy is still lower than *Const.* since nodes in *moving* state basically do not send back a measurement report message. Thus it reduces the redundant messaging in high-mobility situations to mitigate the battery consumption.

4.4.6 Localization Performance in Static Scenarios

We have also evaluated the localization performance in static scenarios where all the nodes are completely stationary. We generated 100 random node topologies by uniformly placing 30 nodes in the field in Fig. 3.4. Then the localization error of the proposed method was compared with that of *Const.* and a belief propagation-based cooperative localization algorithm called SPAWN [39]. In SPAWN, the estimated position of a node is maintained in the form of a probability distribution called *belief*, which is represented by a set of particles. Thus both accuracy and computational cost of SPAWN would depend on the number of particles. To find a reasonable parameter setting for real-time processing on mobile devices, we first implemented SPAWN on the Android platform and evaluated the computation time for *each position update*. In the preliminary experiment, we used the Galaxy Nexus phone and set the number of particles to 4,000, 2,000 or 1,000. As with [39], we assume that the number of particles is reduced by 75% when the belief is exchanged

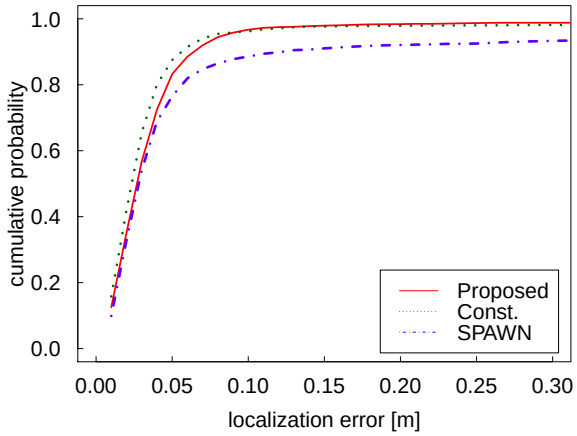


Figure 4.12: Localization performance in static scenarios

between the neighboring nodes to mitigate communication overhead. After 3,000 runs with each configuration, we obtained average computation time of 31.6 seconds, 6.8 seconds and 2.1 seconds, respectively. For real-time tracking of mobile devices, it would be desirable that estimated positions of moving nodes are updated at least every 2 seconds. Therefore we assume that the number of particles is 1,000 and the estimated position is updated at a regular interval of 2 seconds.

Fig. 4.12 shows the cumulative probability of the position errors in 300 seconds from the beginning of the simulations. Due to the limited granularity of the particle-based belief representation, SPAWN performs less than the proposed method. While the positioning accuracy could be improved by increasing the number of particles, it leads to excessive computation time and battery consumption. Instead of maintaining a probability distribution, the proposed method maintains a single point and its "confidence" to simplify the computation. Considering the limited battery and computational resources on mobile devices, it would be preferable that sufficient positioning accuracy can be achieved with minimal resource consumption. The proposed localization component would provide a reasonable solution toward this goal.

4.4.7 Effectiveness of Cooperative Movement Detection

To show the effectiveness of the cooperative movement detection, we also compare its delay time to detect node movement to a non-cooperative method based on the algorithm in [89]. In the non-cooperative method, only fixed anchors transmit measurement signals at regular intervals so that mobile nodes can measure the distance to the anchors. The nodes then calculate variance of the recent measured distances to each anchor and regard itself to be moving if the variance exceeds a pre-defined threshold σ_m^2. While [89] employs received signal strength from WiFi access points, we simply replace it by the ultrasound-based distance measurement for fairness in the evaluation. We assume that the anchors transmit measurement signals every second and the nodes calculate variance of the recent two distance samples. The detection threshold σ_m is set to 0.18m based on the average

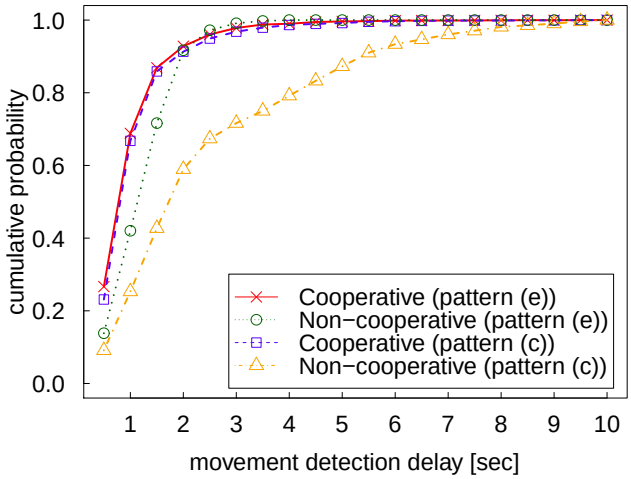


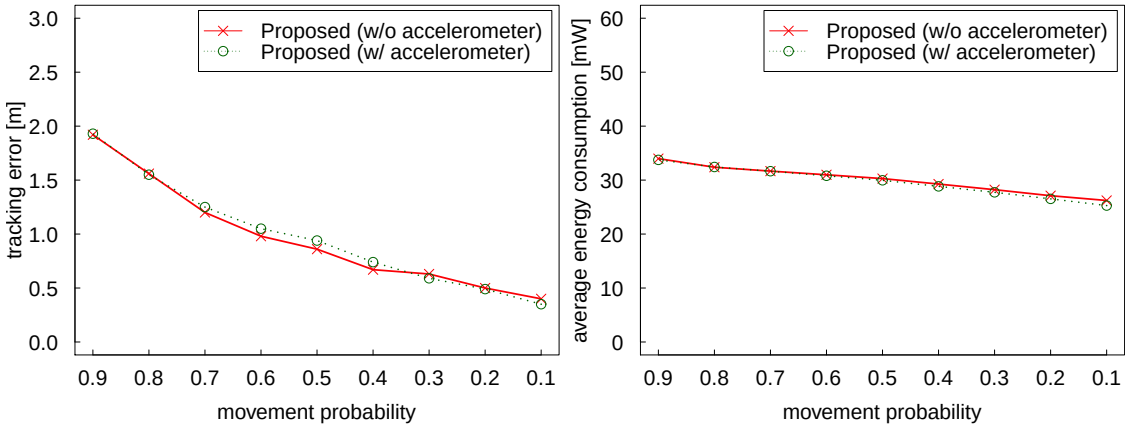
Figure 4.13: Performance of movement detection component

localization error with default parameter settings. Fig. 4.13 shows the movement detection delay of the cooperative and non-cooperative algorithms with two anchor deployment patterns (c) and (e) in Fig. 4.3. The proposed method (*Cooperative*) successfully detects more than 90% of the movements within 2 seconds in both scenarios, while the non-cooperative method tends to take longer delay time especially with the anchor deployment pattern (c). Obviously this difference is due to dependence on fixed anchors; the non-cooperative method can detect movement only when the node is within the measurement range of at least one anchor. On the other hand, the proposed method can cover the areas where the nodes cannot perform distance measurement to any anchors, and thus achieves similar movement detection capability even with the limited anchor coverage.

4.4.8 Movement Detection using Accelerometers

If the mobile devices are equipped with accelerometers, we can also detect movement state of the nodes based on the sensor readings [90, 91]. To evaluate whether the acceleration-based movement detection enhances the tracking performance of the proposed algorithm, we also ran a simulation assuming that movement of a node that is longer than 1.0m can be perfectly detected by the accelerometer within 1 second even if it is not detected by the distance-based cooperative state validation. Fig. 4.14 (a) shows average tracking error when movement probability of the nodes is varied between 0.1 and 0.9. We can see that the tracking performance is almost equivalent regardless of whether accelerometers are used. This is because the movement of a node can be detected by the distance-based state validation mechanism within 1 second in most cases, as we have shown in Fig. 4.13.

We also compare the energy consumption of both cases in Fig. 4.14 (b). With accelerometer-based movement detection, the nodes need not send messages to notify its neighbors of their movement. While it mitigates the communication overhead to some extent, its impact on total energy efficiency is limited because the number of those notification messages is much lower than RTM and measurement report messages. Thus



(a) average tracking errors (b) total energy consumption

Figure 4.14: Performance with accelerometers

contribution of the accelerometers would be limited in terms of movement detection, as far as ranging accuracy is sufficiently high.

4.4.9 Combination with PDR

If inertial sensors are available in the mobile devices, we can also estimate user trajectory by pedestrian dead reckoning (PDR). In this section, we consider the combination with PDR to achieve better tracking performance. Here we assume step-based PDR that is proposed in [76]: User's steps are detected by accelerometer readings, while digital compasses are used to obtain their orientation. Also, stride length of the user is given by a pre-configured parameter. Thus movement vectors of each node can be detected step-by-step and are accumulated in their estimated position. Whenever a new position estimate is obtained by the proposed method, the estimated position of the node is corrected to that position. For the evaluation, we divided the ground truth traces into a sequence of movement vectors with length ρ, where ρ is average step length of the user. Assuming that errors of stride length estimation and orientation estimation follow zero-mean Gaussian distributions $\mathcal{N}(0, \sigma_\rho^2)$ and $\mathcal{N}(0, \sigma_\theta^2)$, respectively, we added a random noise to each movement vector to generate estimated traces by PDR. Note that the parameters are configured as $\rho = 0.58$m, $\sigma_\rho = 0.1$m and $\sigma_\theta = 20^\circ$ based on the experimental results in [76]. We also consider false positive detection of the user's steps due to unexpected motion of the devices. We assume that occurrence of such wrong detection follows a Poisson process with mean interval of 20 seconds and simulate it by a noise vector of length ρ with random orientation. Fig. 4.6 compares the tracking error of the proposed method with/without the PDR-based interpolation. While PDR mitigates the instantaneous errors by updating the estimated position in a timely fashion, error correction should be regularly performed to reset the accumulated errors. Since the proposed method frequently performs localization according to the estimated speed of moving nodes, the accumulated PDR error is immediately corrected, resulting in high tracking performance. We can also see that the

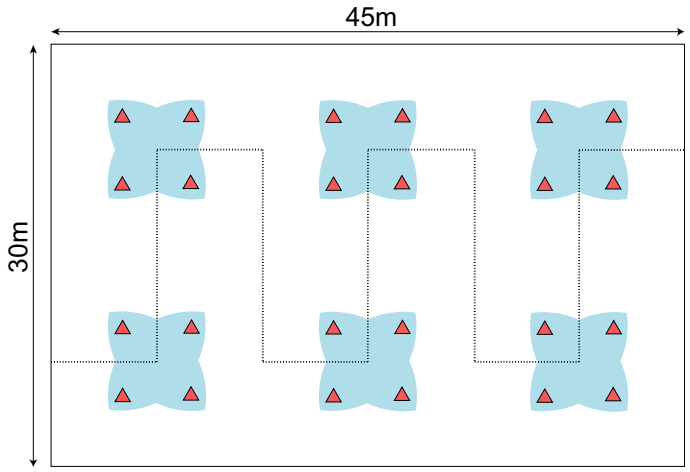


Figure 4.15: Field map

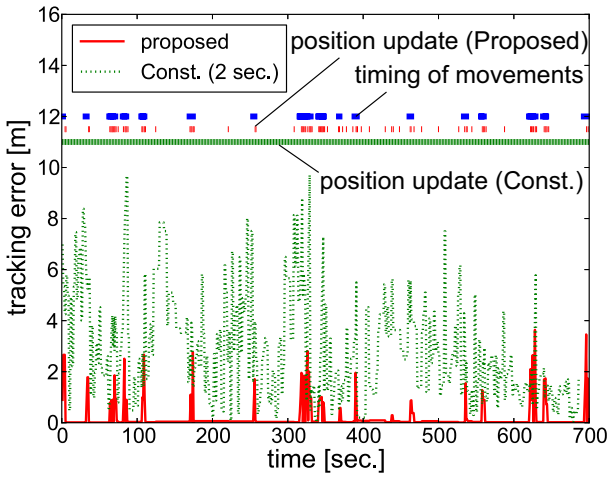


Figure 4.16: Tracking performance in a large space

tracking error with low node density is effectively reduced since the nodes can roughly estimate its position even if the number of neighboring nodes is less than three. Thus utilization of PDR helps to enhance tracking performance, especially in the situations where expected node density is extremely low.

4.4.10 Performance in a Large Space

Finally, we evaluate the performance of the proposed method in a larger space. We assume a 45m×30m field in Fig. 4.15 and 120 mobile nodes that move around the field, following the mobility model in Section 3.3.1. Then a node of interest enters the space from the left boundary and moves toward the opposite side in a stop-and-go manner. Meanwhile, the node moves d meters along the path that is indicated by dotted lines in Fig. 4.15 with probability 0.3, or stays at the current location for 10 seconds with probability 0.7. The distance d of each movement is randomly chosen from 1m to 10m. Fig. 4.16 shows

the timing of position updates and temporal change in tracking errors of the node of interest. For *Const.*, position error continues at a high level due to error propagation from the moving nodes. Conversely, the proposed method keeps reasonable accuracy by the pseudo-anchor selection mechanism, even in the regions that are not covered with a sufficient number of fixed anchors. We can also see that the estimated position is updated only when the node is moving, and thus the total number of localization attempts is effectively reduced.

4.5 Conclusion

In this chapter, we have designed a general framework for performance analysis of the mobility-aware cooperative localization algorithms that exclude the nodes in motion from the set of reference points. In order to clarify when and how they can achieve optimal performance, we have derived the theoretical lower bound of the localization errors. Through a case study, we have compared performance of the proposed localization algorithm in Chapter 3 with the theoretical bound, and shown that the errors asymptotically approach the lower bound as more than 50% of neighboring devices maintain correct movement state. In addition, we have further extended the proposed localization algorithm based on the observations from the theoretical analysis, in order to achieve better positioning accuracy. The framework of error analysis is also applicable to any other cooperative localization algorithms that select pseudo-anchors based on their movement state.

As well as the error analysis, performance of the mobility-aware approaches has been also analyzed from various aspects. The simulation results have suggested an important observation about anchor deployment strategy: Since the proposed method utilizes stopping nodes as virtual anchors (*i.e.* pseudo-anchors), total density of the anchors and stopping mobile nodes dominates the "coverage" of a target field. At each moment, the tracking performance can be kept *regardless of the coverage of fixed anchors* as far as the nodes can obtain the distances to at least three anchors or pseudo-anchors anywhere in the field. Otherwise the localization may fail in the uncovered area, which possibly results in degradation of tracking performance. If expected density of mobile nodes is low or mobility of the nodes seems to be frequent, we should place more anchors to complement the insufficient reference points. Since the success rate of localization is more likely to decline near the boundaries of the field, it may be effective to add some anchors to such near-boundary area. Once we carefully determine the number and deployment of anchors according to the expected density and mobility of mobile nodes, we can effectively take advantage of the proposed method to achieve robust and accurate tracking at a reduced cost. If inertial sensors are available in the mobile devices, combination with PDR techniques would also offer a solution to deal with such high-mobility or low-density situations.

Chapter 5

Mobile Phone Localization Focusing on Collective Activity in Pedestrian Crowds

5.1 Introduction

This chapter further extends the notion of mobility-aware cooperative localization and presents an infrastructure-free approach to relative positioning of nearby pedestrians. The goal of the system is to provide a *local map of the surrounding crowd* in order to underpin socially-aware applications [92] which support social interaction between multiple users. The proposed method, called *PCN* (the acronym of people-centric navigation), employs pedestrian dead reckoning (PDR) to estimate user traces and received signal strength (RSS) of Bluetooth radio for proximity sensing, both of which can be accomplished by off-the-shelf mobile phones. By overlapping the estimated traces at the points where the users seem to have encountered, PCN derives the relative position between the users. To cope with large position errors due to sensor noise, variance of Bluetooth RSS and other environmental factors, we focus on *collective activity* of the people in the crowd. In crowded situations like exhibitions and parties, people often move together with some others, forming a "group." PCN dynamically detects similarity of their activities by gathering mobile phone's acceleration, direction and Bluetooth RSS, and then corrects deviation of the estimated traces by harmonizing with the traces of other group members. Such group-based error correction alleviates the impact of unstableness in the heading estimation, misdetection of user's steps and variance of Bluetooth RSS, which may affect the relative position accuracy. Through a field experiment using Android smartphones, we show that the error correction mechanism successfully enhances positioning accuracy by 28% (from 4.16m to 3.01m). Furthermore, we also analyze the performance of the proposed method in detail through extensive simulations.

The rest of this chapter is organized as follows. Section 5.2 describes observations from our preliminary experiment, followed by basic idea and architecture of the PCN system. Section 5.3 presents detailed design of the proposed localization algorithm, and Section 5.4

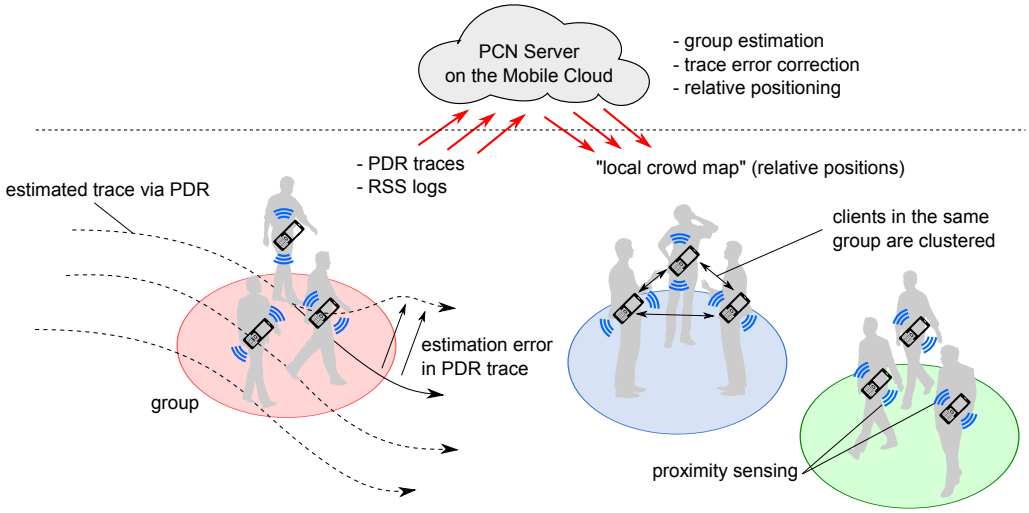


Figure 5.1: PCN system overview

shows how it deals with potential error degradation caused by diversity of device poses. Sections 5.5 and 5.6 provide performance evaluation through our field experiments and extensive simulations, respectively. Then we discuss an example application that can be supported by the proposed method and some related issues in Section 5.7. Finally, Section 5.8 concludes this chapter.

5.2 Overview

5.2.1 System Architecture

The overview of the PCN system is shown in Fig. 5.1. We assume that people who participate in the service run client software on their mobile phone and enable Bluetooth communication. This would be a reasonable assumption in many practical scenarios. For example, organizers of an exhibition can distribute some special mobile application for the patrons to provide information about exhibition contents or congestion at each booth, which motivates people to join the service. In the context of mobile social navigation, attendees in a party would share strong incentive to know where one's acquaintance is. Clients of PCN (*i.e.*, mobile phones) continuously obtain accelerometer and digital compass readings to estimate step counts and heading direction. They also estimate a vector of each step called *step vector* using the direction information and stride length. Since the stride length varies between individuals, it is approximated from the body height. The clients also record RSS from the neighboring clients, which is collected through device discovery process of Bluetooth. The step vectors and RSS are transferred to a centralized server called *PCN server* via 3G or Wi-Fi, and the collected RSS is transformed into distance based on a predefined RSS-to-distance function at server side. Then the PCN server estimates relative positions among the users and the results are sent back to the clients to tell the estimated situation.

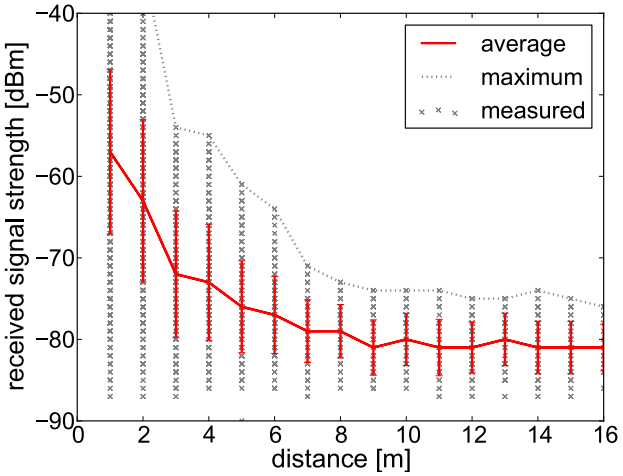


Figure 5.2: Distance vs. Bluetooth RSS

Note that our algorithm is essentially not dependent on the centralized server. If we can locally collect PDR traces and proximity information from the neighboring devices via wireless ad-hoc communication, relative positioning can be done in a distributed manner. For simplicity of discussion, we assume the centralized architecture hereafter.

5.2.2 Impact of Noise on Proximity Sensing and Trace Estimation

As we have stated in the previous sections, step vectors and RSS-based distance information contain non-negligible errors. Fig. 5.2 shows distance-to-Bluetooth RSS mapping based on a real measurement using two Android phones (Nexus S) in our department building receiving a lot of interference from Wi-Fi. We have plotted the measured RSS at each distance and their maximum and average values, where error bars show the standard deviations. As shown in previous literature such as [93], we can see that different RSS values were observed at the same distance due to multipath effect, interference and so on. However, we focus on a criterion to characterize this relation based on the maximum RSS values; at 7m or longer distances they never reach -70dBm, while they exceed it at 6m or shorter distances. We utilize this characteristic to detect "proximity" explained later, allowing some inaccuracy around the boundary of two categories.

We have also implemented a simple PDR application on the Android phones to examine accuracy of step vectors. This application continuously monitors acceleration in the vertical direction to detect steps and the compass readings to estimate the orientation of mobile phones. As shown in Fig. 5.3, the vertical acceleration changes synchronously with the user's steps. Therefore we simply count up the number of steps when the acceleration exceeds a threshold where the minimum counting interval is set to 300 milliseconds to prevent double counting. Using this application, we have analyzed the estimated trace of a user walking twice on the boundary of a 5m$\times$10m rectangle region (see Fig. 5.4). On the estimated trace, the true positions of the three different points highlighted by dotted circles are identical; we can see that the position error grew up to 5.16m on average after

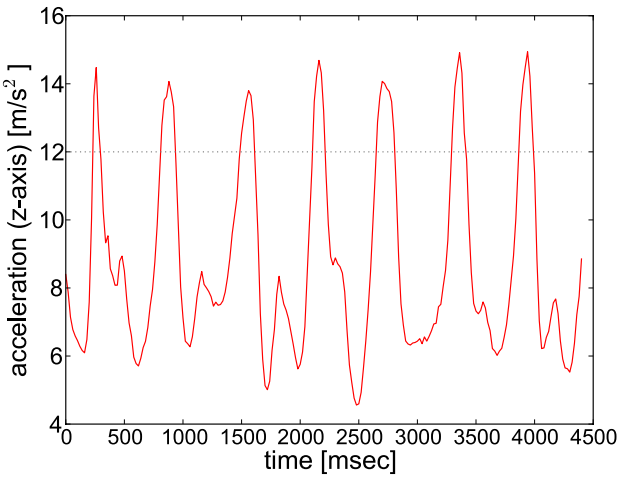


Figure 5.3: Vertical accelerations

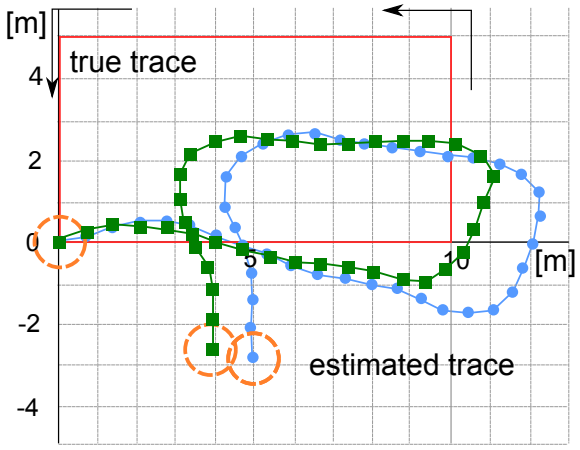


Figure 5.4: Trace deviation in PDR

30m walking. We note that this is the simplest threshold-based PDR where mobile phones are assumed to be held vertically in hand. There are of course more sophisticated PDR methods such as [28, 73]. These methods can be also used to improve PDR accuracy in PCN since it just uses trace estimation results by PDR. For simplicity of discussion, we assume this simple PDR hereafter.

5.2.3 Formal Definition of Groups

Before describing details of the system design, we formally define "groups" in the proposed algorithm. Let S be the set of PCN clients. For a pair of clients $A_i, A_j \in S$, we regard them to have *group relationship* (denoted by $A_i \sim A_j$) if distance between A_i and A_j is continuously less than λ for the last T seconds, where λ and T are parameters depending on mobility characteristics of the users. In the evaluations in Sections 5.5 and 5.6, we set $T = 60$ seconds and $\lambda = 3.0$m.

Figure 5.5: Preliminary experiment

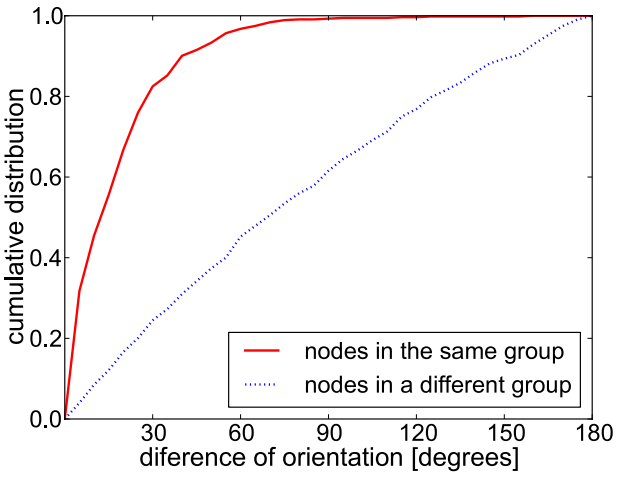


Figure 5.6: Deviation of moving directions

For each client $A_i \in S$, the set of clients that have group relationship with A_i (*i.e.*, $\{A_j | A_j \in S, A_i \sim A_j\}$) is defined to be a *group* which A_i belongs to.

5.2.4 Trace Similarity in a Group

To examine the similarity of traces in a group, we have conducted the following experiment using the PDR application. We let 15 examinees with Nexus S phones freely form groups and let them walk for 30 minutes in a 10m×10m field where markers were placed with one meter spacing. The examinees also took videos of the markers to record true traces, as shown in Fig. 5.5. The obtained traces were broken down into subtraces of 2 seconds, and the average direction of each group was calculated at each time. Then directions of the subtraces were compared to each group average to examine the deviation of orientation within and without the group. Fig. 5.6 shows the cumulative distribution of directional deviation from the group average. The deviation was 30 degrees or less for about 80% of the subtraces in the same group, while it distributes uniformly for those in different groups.

Thus we have confirmed that the traces of users in the same group have highly correlated with each other. It is the basis of the group-based trace error correction explained in the following sections.

5.2.5 Positioning Process of PCN

Positioning process of PCN is composed of three phases. To alleviate position errors due to sensor noise and variance of RSS, at each time step, PCN first estimates group relationship between the clients. It extracts feature values from the recent history of PDR traces and RSS logs, and then calculates the likelihood that each pair of clients belong to the same group. Details of the group estimation algorithm will be described in Section 5.3.2.

Assuming that clients that have kept similar movement behavior for certain duration (*i.e.*, belonging to the same group) continue to have similarity in their trajectories at the next time step, PCN heuristically corrects the original PDR trace to make it approach the average behavior of all the group members. This alleviates the impact of unstableness in the heading estimation and misdetection of user's steps, which may affect relative position accuracy. The algorithm for group-based trace error correction will be presented in Section 5.3.3.

After applying the trace error correction, relative distance between the estimated traces is adjusted based on peer-to-peer Bluetooth RSS that has been collected in the current time step to finally obtain relative position estimates. Since RSS-based distance estimation is poor in accuracy, we also introduce a heuristic for this process; we apply different distance estimation models according to the group relationship between the clients. Based on the assumption of proximity within the group, clients in the same group are attracted to each other more aggressively within the limit of expected variance of RSS values. This helps to alleviate the impact of uncertainty in distance estimation due to attenuation and reflection of radio signals by human bodies and surrounding objects. The algorithm for relative positioning will be described in Section 5.3.4.

Thus, PCN effectively enhances the trace and relative position accuracy without relying on any additional devices, information or infrastructure.

5.3 System Design

5.3.1 Proximity Sensing via Bluetooth Scans

First, we describe the design of proximity sensing. As mentioned above, PCN clients collect Bluetooth RSS from nearby clients via device discovery process, or Bluetooth scan. Since Bluetooth does not have explicit broadcast mechanism, it is the only way to instantly collect peer-to-peer RSS without link management. Ordinary Android phones basically take more than 10 seconds for each attempt of Bluetooth scans, which imposes a severe limitation on the frequency of the scan attempts. To make matters worse, during the process of a scan, the device can hardly respond to the inquiries from other neighbors

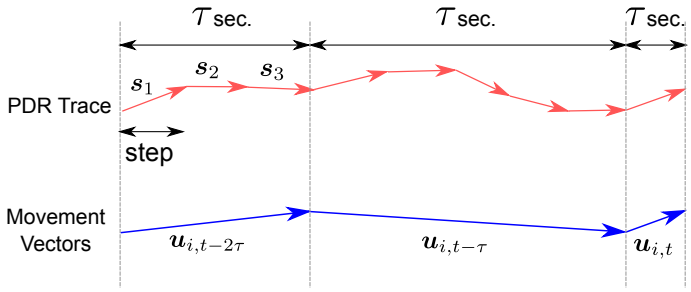


Figure 5.7: Step vectors and movement vectors

because they quickly change their radio frequency meanwhile. Consequently, they often miss nearby clients if they just repeat Bluetooth scans.

In the device discovery process, a device repeatedly transmits inquiry messages for a long period of time so that all the devices in its radio range can be robustly detected. On the other hand, radio signals from nearby clients are relatively free from signal attenuation and thus can be usually detected in a short time. In proximity sensing, quickly detecting the neighbors in close proximity is more important than ensuring exhaustive detection. For that reason, we decided to interrupt each Bluetooth scan in 5 seconds from the beginning. As a result of preliminary experiment, we confirmed that the success rate of scans between nearby clients within 5m is degraded by only about 10% by such interruption, while enhancing the frequency of scans twofold. In addition, we randomly determine the timing of scans to avoid misdetection caused by simultaneous scan attempts. At each time, clients start scanning with probability p_{scan}, or otherwise they sleep for a designated backoff time. In our implementation, we set p_{scan} to 0.5 and randomly pick out a backoff time from 2.5 seconds to 7.5 seconds. Thus we achieve reasonable performance in proximity sensing via Bluetooth.

5.3.2 Group Estimation

At the end of each time step of τ seconds, clients $A_i \in S$ report Bluetooth RSS and PDR traces to the PCN server. The estimated traces by PDR are obtained in the form of a sequence of 2-dimensional step vectors $< \boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_m >$, where each element $\boldsymbol{s}_k$ represents the displacement by a step. Here we define a *movement vector* of each client A_i by the total displacement during the τ seconds, as shown in Fig. 5.7. Let $\boldsymbol{u}_{i,t}$ be the movement vector of A_i at time t in the algorithm descriptions below. Also, we let $r_{ij,t}$ denote average RSS value observed between A_i and $A_j \in S$ during the interval $(t-\tau, t]$. Note that $r_{ij,t}$ is always identical with $r_{ji,t}$ since the RSS values observed by A_i and A_j are both collected to the PCN server.

PCN extracts people's collective activity (*i.e.*, groups) by analyzing the recent PDR traces and measured RSS, which are maintained in the form of fixed length sequences. For each client A_i and time t, let $R_{ij}(t) =< r_{ij,t-(N-1)\tau}, r_{ij,t-(N-2)\tau}, \ldots, r_{ij,t} >$ be the sequence consisting of the RSS samplings from A_j that have been observed during the

last N time steps, and $U_i(t) =< \boldsymbol{u}_{i,t-(N-1)\tau}, \boldsymbol{u}_{i,t-(N-2)\tau}, \ldots, \boldsymbol{u}_{i,t} >$ be the sequence of its recent N movement vectors. Here, we set the unit time τ to 2 seconds and the window size N to 30, respectively.

We characterize the likelihood that a pair of clients A_i and $A_j \in S$ belong to the same group (*i.e.*, *group likelihood*) by the following two features;

(i) **Proximity**: The number of RSS samplings larger than a threshold Θ_{prox} which are observed between A_i and A_j during the recent N time steps:

$$n_{ij}(t) = \left|\{r_{ij,t'} | r_{ij,t'} \in R_{ij}(t), r_{ij,t'} > \Theta_{prox}\}\right| \tag{5.1}$$

where Θ_{prox} is given as a system parameter.

(ii) **Trace similarity**: Edit distance [94] between their sequences of recent movement vectors over the recent N time steps:

$$d_{ij}(t) = ED(U_i(t), U_j(t)) \tag{5.2}$$

In general, edit distance between two sequences U and V is defined by the number of insert, delete and replace operations that are needed to convert U into V. We regard corresponding movement vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ to be matched if they satisfy both of the following criteria:

$$|\ ||\boldsymbol{u}|| - ||\boldsymbol{v}||\ | < \Theta_l, \quad |\arg(\boldsymbol{u}) - \arg(\boldsymbol{v})| < \Theta_\theta \tag{5.3}$$

where Θ_l and Θ_θ are matching thresholds. Note that we skip the angular criterion if $\boldsymbol{u} = \boldsymbol{0}$ or $\boldsymbol{v} = \boldsymbol{0}$ since we cannot define orientation of the vector in such cases. Now let U and V be the sequences of movement vectors, where the length of each sequence is n and m, respectively. We define the edit distance between U and V by the following recursive formula:

$$ED(U,V) = \begin{cases} n \cdot w & \text{if } m = 0 \\ m \cdot w & \text{if } n = 0 \\ \min\{ED(Rest(U), Rest(V)) + c, & \\ \quad ED(Rest(U), V) + w, & \\ \quad ED(U, Rest(V)) + w\} & otherwise \end{cases} \tag{5.4}$$

where $Rest(U)$ represents a sub-sequence of U without the first element. c is a cost function of a replace operation where $c = 0$ if the first element of U and that of V satisfy the matching criteria, or $c = 1$ otherwise. Insert and delete operations correspond to temporal shifting on user traces. We let w denote the cost of such a shift operation, and set it to 0.5. Since there could be a small difference in timing of movements even if A_i and A_j belong to the same group, we intend to mitigate the impact of such temporal variations by imposing a smaller cost value to insert and delete operations.

We represent the group relationship between clients A_i and A_j by a binary random variable G_{ij}, where $G_{ij} = 1$ if $A_i \sim A_j$, or $G_{ij} = 0$ otherwise. Also, we denote $n_{ij}(t)$ and $d_{ij}(t)$ by just n_{ij} and d_{ij} in the following descriptions.

Probability distribution of G_{ij} under given measurements n_{ij} and d_{ij} can be derived by the well-known Bayes' rule:

$$P(G_{ij}|n_{ij}, d_{ij}) = \frac{P(n_{ij}, d_{ij}|G_{ij}) \cdot P(G_{ij})}{\sum_{G_{ij}=0}^{1} P(n_{ij}, d_{ij}|G_{ij}) \cdot P(G_{ij})} \tag{5.5}$$

For simplicity, we assume that n_{ij} and d_{ij} are independent under given G_{ij}:

$$P(n_{ij}, d_{ij}|G_{ij}) = P(n_{ij}|G_{ij}) \cdot P(d_{ij}|G_{ij}) \tag{5.6}$$

Here we also assume that there is no prior information about group relationship between A_i and A_j, that is, $P(G_{ij} = 0) = P(G_{ij} = 1) = 0.5$. Under these assumptions, Eq.(5.5) can be simplified as:

$$P(G_{ij}|n_{ij}, d_{ij}) = \frac{P(n_{ij}|G_{ij}) \cdot P(d_{ij}|G_{ij})}{\sum_{G_{ij}=0}^{1} P(n_{ij}|G_{ij}) \cdot P(d_{ij}|G_{ij})}. \tag{5.7}$$

Distributions $P(n_{ij}|G_{ij})$ and $P(d_{ij}|G_{ij})$ in Eq.(5.7) can be obtained by a preliminary learning (based on either a simulation or a simple field experiment), which will be discussed in Section 5.5.1.

To find a group G_i to which a client $A_i \in S$ belongs, PCN server first calculates n_{ij} and d_{ij} for all the clients $A_j \in S$ based on the recent history of PDR traces and RSS logs. Then it determines group likelihood $P(G_{ij} = 1|n_{ij}, d_{ij})$ by the group estimation model in Eq.(5.7). If the group likelihood between A_i and A_j exceeds a threshold Θ_{group}, A_i and A_j are regarded to have group relationship. PCN forms a set of clients that seem to have group relationship with A_i, and finally regards it as a group to which A_i belongs:

$$G_i = \{A_i\} \cup \{A_j | A_j \in S, P(G_{ij} = 1|n_{ij}, d_{ij}) > \Theta_{group}\} \tag{5.8}$$

We set the threshold $\boldsymbol{\Theta}_{group} = 0.8$ in the following evaluations.

5.3.3 Group-based Trace Error Correction

In this section, we describe the detailed design of our trace error correction mechanism. As mentioned in Section 5.2.5, the correction algorithm is grounded on the assumption that clients having kept similar movement behavior for certain duration are expected to continue similar behavior at the next time step. Thus, PCN heuristically corrects the current movement vector to make it approach the average behavior of the group members within the limit of expected PDR errors included in the original movement vector.

For that purpose, we first construct an error model of PDR to derive error distribution of the original movement vectors. Then we also define an empirical model regarding similarity in trajectories among group members. Finally, we describe how to correct potential errors in movement vectors based on the expected error distribution and the empirical trace similarity model.

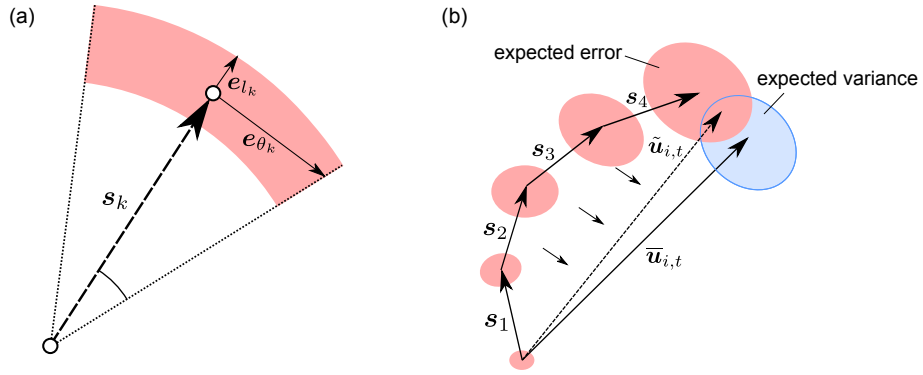


Figure 5.8: Trace error correction: (a) step-level error model, and (b) correcting a movement vector $\boldsymbol{u}_{i,t}$ according to the average vector $\overline{\boldsymbol{u}}_{i,t}$ of the group.

Predicting PDR Error

To derive expected error distribution of the original movement vector, we first model the errors in PDR traces of individual users.

Assuming that a client A_i has detected m steps during the last τ seconds, the movement vector $\boldsymbol{u}_{i,t}$ can be denoted by $\boldsymbol{u}_{i,t} = \sum_{k=1}^{m} \boldsymbol{s}_k$, where $\boldsymbol{s}_k$ is the step vector by the k-th step. PDR error is mainly caused by i) error of step length estimation, ii) error of direction estimation, and iii) misdetection of user steps. Errors caused by i) and ii) occur in every step $\boldsymbol{s}_k$, and accumulated in the movement vector. The expected error in the step $\boldsymbol{s}_k$ can be denoted as $\boldsymbol{e}_k = \boldsymbol{e}_{l_k} + \boldsymbol{e}_{\theta_k}$, where $\boldsymbol{e}_{l_k}$ is the error of estimated step length, say l, and $\boldsymbol{e}_{\theta_k}$ is the error caused by directional distortion (see Fig. 5.8 (a)). Based on a preliminary experiment, we have confirmed that errors of step length and direction estimation almost follow Gaussian distributions. Thus we assume that $\boldsymbol{e}_{l_k}$ is in the same direction as $\boldsymbol{s}_k$ and its length follows Gaussian distribution $\mathcal{N}(0, \sigma_l^2)$. For $\boldsymbol{e}_{\theta_k}$, we assume that direction estimation error $\Delta\theta$ follows Gaussian distribution $\mathcal{N}(0, \sigma_\theta^2)$, and approximate $\boldsymbol{e}_{\theta_k}$ by a vector which is orthogonal to $\boldsymbol{s}_k$ with a length of $l\Delta\theta$. Under this modeling, we can uniquely determine the error distribution of each step vector $\boldsymbol{s}_k$. We empirically set the model parameters σ_l and σ_θ to 0.5 m and 30.0 degrees, respectively.

The error caused by iii) is due to irregular motion of the devices, and thus occurs regardless of the number of user's steps over the last τ seconds. We treat this error (denoted by $\boldsymbol{e}_0$) independent of the step-level error prediction model defined above and model it by a Gaussian distribution in Eq. (5.9):

$$P(\boldsymbol{e}_0) = \mathcal{N}(\boldsymbol{0}, \sigma_0^2 \boldsymbol{I}) \tag{5.9}$$

where $\boldsymbol{I}$ is a 2-dimensional identity matrix. We empirically set σ_0 to 1.0 m.

Using the constructed step-level error prediction model, we estimate the error distribution of a movement vector as follows. As mentioned above, PDR error is accumulated in the movement vector step-by-step as shown in Fig. 5.8 (b). Let $\boldsymbol{u}'_k$ be a partial movement vector composed of $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k$. We recursively predict the distribution of $\boldsymbol{u}_{i,t}$

by sequentially deriving the distributions of $\boldsymbol{u}'_k$ $(k = 0, 1, 2, \ldots, m)$, namely $P(\boldsymbol{u}'_k|\boldsymbol{s}_{1:k})$. Since the error caused by misdetection of user steps could occur regardless of the users' movement, we apply the distribution in Eq. (5.9) as an initial distribution:

$$P(\boldsymbol{u}'_0) = P(\boldsymbol{e}_0). \tag{5.10}$$

Given the distribution at the $(k-1)$-th step (*i.e.*, $P(\boldsymbol{u}'_{k-1}|\boldsymbol{s}_{1:\,k-1})$), we can derive the distribution at the next step using the step-level error prediction model. Relationship between $\boldsymbol{u}'_{k-1}$ and $\boldsymbol{u}'_k$ can be represented as follows:

$$\boldsymbol{u}'_k = \boldsymbol{u}'_{k-1} + \boldsymbol{s}_k + \boldsymbol{e}_k \tag{5.11}$$

where $\boldsymbol{e}_k$ is the error vector included in $\boldsymbol{s}_k$. According to the recursion formula, the updated distribution $P(\boldsymbol{u}'_k|\boldsymbol{s}_{1:\,k})$ can be derived from the previous distribution $P(\boldsymbol{u}'_{k-1}|\boldsymbol{s}_{1:\,k-1})$ and step-level error distribution $P(\boldsymbol{e}_k|\boldsymbol{s}_k)$:

$$P(\boldsymbol{u}'_k|\boldsymbol{s}_{1:\,k}) = \int \left\{ \int P(\boldsymbol{u}'_k|\boldsymbol{s}_k, \boldsymbol{e}_k, \boldsymbol{u}'_{k-1}) \cdot P(\boldsymbol{u}'_{k-1}|\boldsymbol{s}_{1:\,k-1}) d\boldsymbol{u}'_{k-1} \right\} \cdot P(\boldsymbol{e}_k|\boldsymbol{s}_k) d\boldsymbol{e}_k \tag{5.12}$$

Here, we sample N_p particles $\boldsymbol{u}'^{(j)}_{k-1}$ $(j = 1, 2, \ldots, N_p)$ from the previous distribution $P(\boldsymbol{u}'_{k-1}|\boldsymbol{s}_{1:\,k-1})$ to represent it by Monte Carlo approximation:

$$P(\boldsymbol{u}'_{k-1}|\boldsymbol{s}_{1:\,k-1}) \simeq \frac{1}{N_p} \sum_{j=1}^{N_p} \delta(\boldsymbol{u}'_{k-1} - \boldsymbol{u}'^{(j)}_{k-1}) \tag{5.13}$$

By this particle-based representation, the updated distribution in Eq. (5.12) can be transformed into:

$$P(\boldsymbol{u}'_k|\boldsymbol{s}_{1:\,k}) = \frac{1}{N_p} \sum_{j=1}^{N_p} \left[\int P(\boldsymbol{u}'_k|\boldsymbol{s}_k, \boldsymbol{e}_k, \boldsymbol{u}'^{(j)}_{k-1}) \cdot P(\boldsymbol{e}_k|\boldsymbol{s}_k) d\boldsymbol{e}_k \right]. \tag{5.14}$$

For each particle $\boldsymbol{u}'^{(j)}_{k-1}$, we sample a single particle $\boldsymbol{e}^{(j)}_k$ from the step-level error distribution $P(\boldsymbol{e}_k|\boldsymbol{s}_k)$ to approximate it as Eq. (5.15). Note that we sample an error value *for each of N_p particles* to reasonably approximate the step-level error distribution $P(\boldsymbol{e}_k|\boldsymbol{s}_k)$ overall.

$$P(\boldsymbol{e}_k|\boldsymbol{s}_k) \simeq \delta(\boldsymbol{e}_k - \boldsymbol{e}^{(j)}_k). \tag{5.15}$$

By this approximation, Eq. (5.14) can be transformed into:

$$\begin{aligned} P(\boldsymbol{u}'_k|\boldsymbol{s}_{1:\,k}) &\simeq \frac{1}{N_p} \sum_{j=1}^{N_p} \left[\int P(\boldsymbol{u}'_k|\boldsymbol{s}_k, \boldsymbol{e}_k, \boldsymbol{u}'^{(j)}_{k-1}) \cdot \delta(\boldsymbol{e}_k - \boldsymbol{e}^{(j)}_k) d\boldsymbol{e}_k \right] \\ &= \frac{1}{N_p} \sum_{j=1}^{N_p} \left[\delta\left(\boldsymbol{u}'_k - (\boldsymbol{u}'^{(j)}_{k-1} + \boldsymbol{s}_k + \boldsymbol{e}^{(j)}_k) \right) \right]. \end{aligned} \tag{5.16}$$

Based on the discussion above, we design the error prediction algorithm as follows. First, we sample N_p particles $\boldsymbol{u}'^{(j)}_0$ $(j = 1, 2, \ldots, N_p)$ from $P(\boldsymbol{u}'_0)$ in Eq. (5.10). Then, for

each step $\boldsymbol{s}_k$, we add $\boldsymbol{s}_k$ and $\boldsymbol{e}_k^{(j)}$, which is an error value sampled from $P(\boldsymbol{e}_k|\boldsymbol{s}_k)$, to each particle $\boldsymbol{u'}_k^{(j)}$. After repeating this operation for all the steps $\boldsymbol{s}_k$ in the period of recent τ seconds, we can obtain the probability distribution $P(\boldsymbol{u}_{i,t})$ of the movement vector in the form of Monte Carlo representation as $P(\boldsymbol{u}_{i,t}) = P(\boldsymbol{u'}_m|\boldsymbol{s}_{1:\,m})$. We set N_p to 500 in the experiments in Sections 5.5 and 5.6.

Modeling Trace Similarity within a Group

After extracting groups, PCN corrects errors in the *current* movement vectors by harmonizing the expected error distribution of the original movement vector with an empirical distribution given by the assumption of trace similarity within a group. The former distribution is derived from the error model of PDR, which has been constructed in the previous section. To determine the latter distribution, for each client A_i, PCN first identifies typical behavior of its group G_i by calculating "average" movement vector of the group members. Due to measurement noise and irregular motion of the users, it may often happen that movement vectors of some clients in G_i substantially deviate from other group members. To prevent such deviated traces from affecting trace accuracy of other group members, PCN excludes the isolated traces in calculating the average behavior of the group. We find such deviated traces by distance-based local outlier detection [95]. With this scheme, a movement vector $\boldsymbol{u}_{j,t}$ of a client $A_j \in G_i$ is regarded as a $DB(p, D)$-*outlier* if at least a fraction p of movement vectors in G_i lie greater than distance D from $\boldsymbol{u}_{j,t}$. By default, we assign $p = 0.5$ and $D = \overline{d}$, respectively, where $\overline{d}$ is the average Euclidean distance between the movement vectors of the clients in G_i. If a majority of the clients fall into $DB(0.5, \overline{d})$-*outlier*, we alleviate the criterion by increasing p and try the detection process again until more than a half of the group members remain non-outliers. Given that $G'_i \subseteq G_i$ is the set of clients that have survived the outlier detection, we let $\overline{\boldsymbol{u}}_{i,t} = \frac{1}{|G'_i|}\sum_{A_j \in G'_i} \boldsymbol{u}_{j,t}$ represent the typical behavior (*i.e.*, *group average*) of the group G_i. Then, for each movement vector $\boldsymbol{u}_{i,t}$ and its corresponding group average $\overline{\boldsymbol{u}}_{i,t}$, we compare the length and direction of $\boldsymbol{u}_{i,t}$ and $\overline{\boldsymbol{u}}_{i,t}$. To model the trace-similarity-based distribution, we assume that the ratio $(||\boldsymbol{u}_{i,t}|| - ||\overline{\boldsymbol{u}}_{i,t}||)/||\overline{\boldsymbol{u}}_{i,t}||$ follows a Gaussian distribution $\mathcal{N}(0, \sigma^2_{g_l})$ and $(\arg(\boldsymbol{u}_{i,t}) - \arg(\overline{\boldsymbol{u}}_{i,t}))$ follows $\mathcal{N}(0.0, \sigma^2_{g_\theta})$, respectively. Based on the result of our preliminary experiment, we set the parameters σ_{g_l} and σ_{g_θ} to be 0.50 and 30.0 degrees, respectively. Since we found that the distribution of $(||\boldsymbol{u}_{i,t}|| - ||\overline{\boldsymbol{u}}_{i,t}||)$ varies with moving speed, we normalize it with $||\overline{\boldsymbol{u}}_{i,t}||$ to alleviate the impact of users' behavior. We define the empirical model regarding similarity in trajectories within a group, namely, $P(\boldsymbol{u}_{j,t}|\overline{\boldsymbol{u}}_{i,t})$ by the product of these two distributions.

Correcting PDR Traces

Finally, likelihood distribution of the movement vector is updated by multiplying the error distribution $P(\boldsymbol{u}_{i,t})$ of the original movement vector by the trace-similarity-based distribution $P(\boldsymbol{u}_{i,t}|\overline{\boldsymbol{u}}_{i,t})$. Weighting each particle $\boldsymbol{u}_{i,t}^{(j)}$ with the trace-similarity-based distribution

$P(\boldsymbol{u}_{i,t}^{(j)}|\overline{\boldsymbol{u}}_{i,t})$, we recalculate the expected value of $\boldsymbol{u}_{i,t}$ and adopt it as the corrected movement vector $\tilde{\boldsymbol{u}}_{i,t}$:

$$\tilde{\boldsymbol{u}}_{i,t} = \sum_{j=1}^{N_p} \boldsymbol{u}_{i,t}^{(j)} \cdot \frac{P(\boldsymbol{u}_{i,t}^{(j)}|\overline{\boldsymbol{u}}_{i,t})}{\sum_{k=1}^{N_p} P(\boldsymbol{u}_{i,t}^{(k)}|\overline{\boldsymbol{u}}_{i,t})}. \tag{5.17}$$

What we essentially do in Eq. (5.17) is to multiply two different distributions of $\boldsymbol{u}_{i,t}$ together, normalize the product so that it becomes a valid probability distribution function (PDF), and compute the mean of the new PDF to derive the corrected movement vector.

5.3.4 Determining Relative Positions

In the final phase, PCN estimates and updates relative positions of the clients based on the corrected movement vectors $\{\tilde{\boldsymbol{u}}_{i,t}|A_i \in S\}$ and Bluetooth RSS $\{r_{ij,t}|A_i, A_j \in S\}$ that have been collected in the current time step.

The position estimation is basically grounded on "encounters" of the clients like [12, 38]. Let $\boldsymbol{p}_{ij,t}$ be the estimated relative position of a client A_j with respect to a client A_i at time t, and $\delta_{ij,t}$ be its uncertainty. Initially, the relative position is *unknown* for all the pairs of clients. Then at each time step, PCN system concludes that the clients A_i and A_j encountered if $r_{ij,t}$ is larger than the threshold $\boldsymbol{\Theta}_{prox}$. If $\boldsymbol{p}_{ij,t}$ is still *unknown* when their encounter is detected, relative position between A_i and A_j is initialized by $(0, 0)$. Given that RSS-based distance between A_i and A_j is $D_{ij,t}$, actual relative position of A_j with respect to A_i would be on the circumference with a radius of $D_{ij,t}$ centered at A_i. We define the uncertainty of $\boldsymbol{p}_{ij,t}$ by the maximum distance between the estimated position and the circumference, namely, $\delta_{ij,t} = D_{ij,t}$.

Afterwards, the estimated position is updated by the corrected movement vectors at every time step:

$$\boldsymbol{p}_{ij,t} = \boldsymbol{p}_{ij,t-\tau} - \tilde{\boldsymbol{u}}_{i,t} + \tilde{\boldsymbol{u}}_{j,t}. \tag{5.18}$$

On updating the estimated position $\boldsymbol{p}_{ij,t}$, we also update the corresponding uncertainty $\delta_{ij,t}$ by adding the standard deviation of error distributions calculated for each movement vector $\tilde{\boldsymbol{u}}_{i,t}$ and $\tilde{\boldsymbol{u}}_{j,t}$.

If A_i and A_j have already encountered and PCN maintains their relative positions, $r_{ij,t}$ is utilized to adjust their relative distance. By referring to the pre-configured RSS-to-distance mapping with $r_{ij,t}$, PCN estimates relative distance $D_{ij,t}$ between the clients A_i and A_j. If $||\boldsymbol{p}_{ij,t}||$ is larger than $D_{ij,t}$, we correct $\boldsymbol{p}_{ij,t}$ so that it satisfies the distance constraint:

$$\boldsymbol{p}'_{ij,t} = \boldsymbol{p}_{ij,t} \cdot \frac{D_{ij,t}}{||\boldsymbol{p}_{ij,t}||}. \tag{5.19}$$

In this case, the uncertainty of $\boldsymbol{p}_{ij,t}$ is updated to $\delta_{ij,t} = \min(\delta_{ij,t-\tau}, 2D_{ij,t})$. Note that the new uncertainty is $2D_{ij}$ rather than D_{ij} due to flip ambiguity.

As discussed in Section 5.2, the large deviation of measured RSS imposes a severe limitation to the accuracy of distance estimation. To pursue finer-grained positioning, here we also inspire a heuristic based on the assumption of proximity within a group. Recall the RSS measurement experiment in Section 5.2. Two clients, say A_1 and A_2, are

placed d meters apart and then one of the clients A_1 repeatedly performs Bluetooth scans N_r times. If A_1 fails to detect A_2 in a scan, the minimum RSS value (*i.e.*, -99 dBm) is assigned to the corresponding sampling. Let $r_u(d)$ and $r_l(d)$ be the u-th and l-th ($u < l$) largest RSS samples observed at distance d, respectively. Based on the measurement results, we define two RSS-based distance estimation models in Eq. (5.20):

$$D_{ij,t} = \begin{cases} d_0 \quad s.t. \quad r_{ij,t} = r_u(d_0) & \text{if } G_{ij} = 0 \\ d_1 \quad s.t. \quad r_{ij,t} = r_l(d_1) & \text{if } G_{ij} = 1 \end{cases} \tag{5.20}$$

In the distance adjustment, PCN chooses a proper model according to the group relationship between the clients. Consequently, clients in the same group are attracted to each other more aggressively within the limit of expected variance of RSS values. It may alleviate the impact of uncertainty in distance estimation due to attenuation and reflection of radio signals at human bodies and surrounding objects. In this dissertation, we set $u = 0.25N_r$ and $l = 0.75N_r$, respectively.

Now we consider the situation that a client A_1 knows relative position to A_2 and A_2 knows relative position to A_3, while A_1 and A_3 have never encountered directly. In this case, relative position between A_1 and A_3 can be roughly estimated by adding $\boldsymbol{p}_{1,2}$ and $\boldsymbol{p}_{2,3}$. To do this more systematically, we construct a *relative position graph* as follows: We first generate vertexes corresponding to the clients in S. Then an edge is created between the clients that have encountered each other and thus their relative position $\boldsymbol{p}_{ij,t}$ is maintained by PCN. Each edge is associated with the relative position and its uncertainty $\delta_{ij,t}$. PCN derives relative position between each pair of clients by finding the shortest path on the relative position graph, where the uncertainty serves as a weight of each edge. Thus, the final position estimate can be derived by adding all the hop-by-hop relative positions on the path.

5.4 Dealing with Different Phone Poses

In the PDR algorithm in Section 5.2.2, we assumed that users always hold their mobile device in hand. However, in practical situations, people often carry their phone in different manners (*e.g.*, in a pocket or in a bag) and frequently change the device placement. In this section, we extend our algorithm to alleviate the assumption regarding the pose of the client device.

To deal with different phone poses, we first analyze the following two potential issues; (i) impact of device placement on the reception of Bluetooth signals and (ii) difference in error characteristics of the PDR traces according to the device pose. To observe the characteristics of the Bluetooth signal reception with different phone poses, we have conducted the following experiment using two Nexus S phones: One of the phones is placed at a distance of d ($1\text{m} \leq d \leq 20\text{m}$), listening to the inquiries from neighboring phones. Then a person holds the other phone with the following four poses and repeatedly performs Bluetooth scans:

(i) *hand* (*front*): The user holds the phone in hand, facing the other phone.

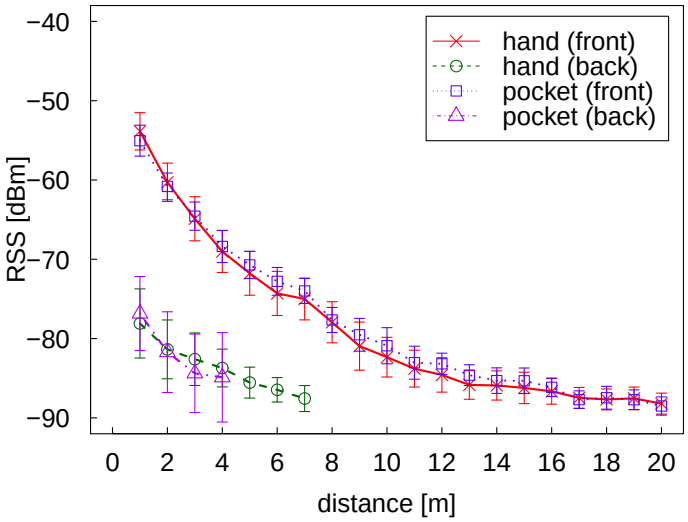


Figure 5.9: Bluetooth RSS with different phone poses

(ii) *hand* (*back*): The phone is also held in hand, while the user faces the opposite direction.

(iii) *pocket* (*front*): The user puts his phone into the front trouser pocket, facing the other phone.

(iv) *pocket* (*back*): The user puts his phone in the front trouser pocket, facing the opposite direction.

Note that the experiment was held in outdoor open space to clearly observe the impact of the phone poses, minimizing the effect of other environmental factors such as multipath propagation and interference from WiFi. Fig. 5.9 shows the mean and standard deviation of Bluetooth RSS observed at each distance. From the results, we can see that the RSS values with *pocket* (*front*) are almost equivalent to those with *hand* (*front*). On the other hand, RSS values steeply decline and rarely exceed the threshold of −70dBm when the radio signals are obstructed by a human body (*i.e.*, with *hand* (*back*) and *pocket* (*back*)). Thus the impact on the Bluetooth signal reception essentially depends on whether the signal propagation path is obstructed by human bodies. In that sense, the *in-pocket* cases are more likely to be affected by the signal attenuation since the position of the client device is closer to the user body, and thus radio signals from the backward of the user are usually obstructed.

As well as the Bluetooth signal reception, difference in phone poses would also affect characteristics of the PDR errors. For example, we consider the step-based PDR algorithm described in Section 5.2. If the user puts the phone into a trouser pocket, there may be some offset between directions of the user heading and the device heading. If gyro sensors are available in the mobile devices, the initial direction offset can be estimated and corrected by tracking the pose of the device while the user puts the phone from hand into the pocket [28]. Nevertheless, the PDR errors may become larger than the *in-hand*

case since the device would frequently jump and move in the pocket during the walking motion. In order to observe the difference in characteristics of the PDR errors between the device poses, we have also conducted the following preliminary experiment: Three student volunteers walked in outdoor space holding two Nexus S phones. One of the phones was held in hand and the other was put in the front trouser pocket. During the experiment, the PDR application used in the field experiment was run on both phones to collect estimated step vectors. In addition, location of the phone was continuously obtained by GPS to collect ground truth traces. Based on the experiment, we have found that variance of the direction estimation errors with the *in-pocket* device is larger by 23^2 degrees compared to the *in-hand* case. For PDR traces with the *in-pocket* devices, we assume that direction estimation errors follow a Gaussian distribution $\mathcal{N}(0, \sigma_\theta^2 + \sigma_{\delta\theta}^2)$ instead of $\mathcal{N}(0, \sigma_\theta^2)$. The standard deviation of additional direction errors is given by $\sigma_{\delta\theta} = 23$ degrees based on the preliminary experiment above.

To effectively reduce the trace estimation errors by the group-based trace correction mechanism, we need to apply an appropriate PDR error model according to the pose of the client device. Park et al. [96] propose an algorithm to classify device placement (*i.e.*, in hand, at ear, in trouser pocket or in a backpack) with 94% accuracy using an accelerometer in the mobile device. Utilizing such online pose estimation techniques, we can extend the proposed method to cope with the situations where the users dynamically change the device placement: We assume that current pose of the device is monitored by the client (*e.g.*, using the accelerometer-based pose estimation algorithm in [96]) and reported to the PCN server. In deriving the error distribution of the movement vectors, the server dynamically switches the parameters of the PDR error model according to the estimated device pose.

Another concern is diversity of device placement among the clients. The group-based trace error correction algorithm in Section 5.3.3 calculates the average movement vector of all the group members and then corrects the movement vector of the client such that it approaches the resulting group average. If the expected errors in the movement vectors vary among the group members, clients with large trace errors (*i.e.*, those who carry the device in a pocket) may negatively impact the trace estimation accuracy of the other members who hold their phone in hand. An approach to mitigating such error propagation is to consider confidence of the movement vectors in calculating the group average. For this purpose, we define the confidence c_i of each client A_i based on the variance of the resulting trace error distribution, which is estimated using the PDR error model corresponding to the reported device pose:

$$c_i = \frac{1}{\sqrt{\frac{\sum_{j=1}^{N_p} ||\boldsymbol{u}_{i,t}^{(j)} - \boldsymbol{u}_{i,t}||^2}{N_p - 1}}}. \tag{5.21}$$

Weighting the movement vectors of each member with their confidence, group average $\overline{\boldsymbol{u}}_{i,t}$ is calculated as:

$$\overline{\boldsymbol{u}}_{i,t} = \frac{\sum_{A_j \in G_i'} c_j \cdot \boldsymbol{u}_{j,t}}{\sum_{A_j \in G_i'} c_j} \tag{5.22}$$

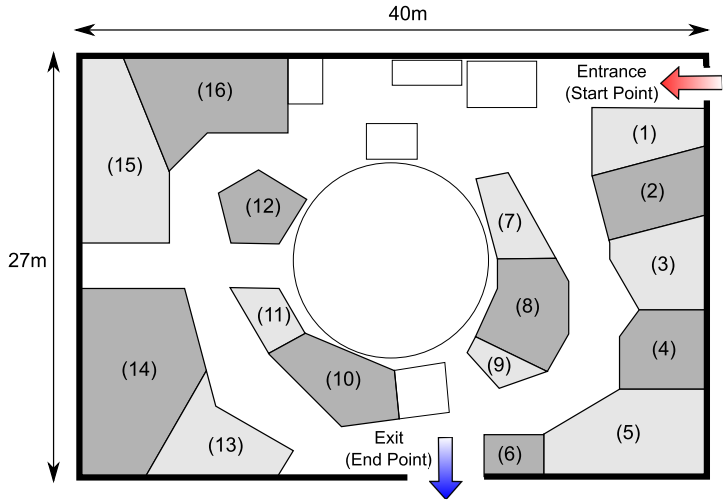


Figure 5.10: Floor map

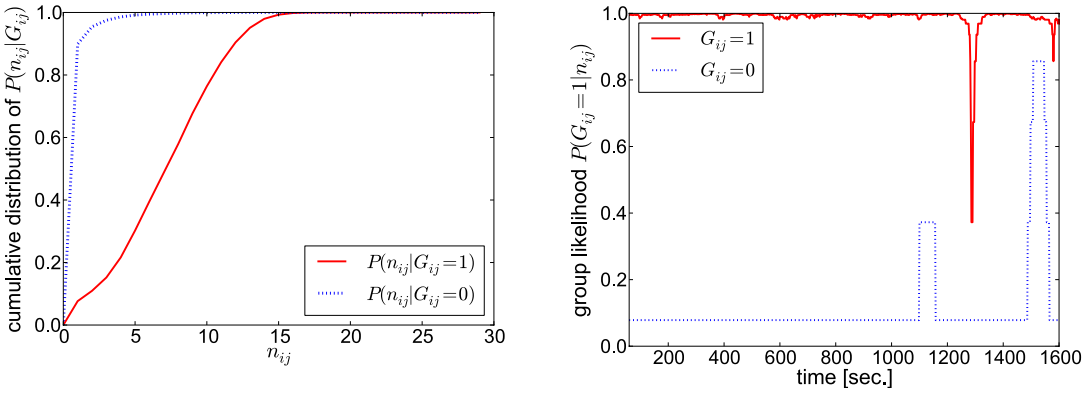
Figure 5.11: Field experiment

where G'_i is the set of group members of the client A_i after the outlier elimination in Section 5.3.3.

5.5 Evaluation

To collect sensor data and RSS logs in real environment, we conducted a field experiment in a public trade fair. As part of a technical event named Knowledge Capital Trial 2011 (`http://www.kmo-jp.com/en/`), a trade fair was held at a 27m×40m-sized hall as shown in Fig. 5.10. Totally 16 industrial companies and universities exhibited their state-of-the-art technology while thousands of visitors went around the booths.

We let 20 students hold Nexus S phones and asked to go around the event place with a group of four people. A sensing application equipped with PDR and proximity sensing was running on each phone to record users' trace and RSS logs. Each group entered the event place at the entrance, and then looked around 6-12 booths for about 30 minutes. After that, they left there through the exit shown in Fig. 5.10. Usually they stayed at



(a) distribution of n_{ij} (b) group likelihood (proximity)

Figure 5.12: Proximity-based group classification model

each booth for 1-5 minutes on average. To collect ground truth data of user traces, we assigned an additional person to each group to plot their true positions with timestamps on a field map when arriving at and leaving a booth, as well as taking their photos at certain time intervals. After conducting such experiments three times, we collected real sensor data and RSS logs that are about 1,800 minutes long in total (90 minutes logs for 20 examinees). In the following evaluation, we used the logs from two experiments as a learning dataset to construct a group classifier, and the remaining one as a test dataset to examine the performance.

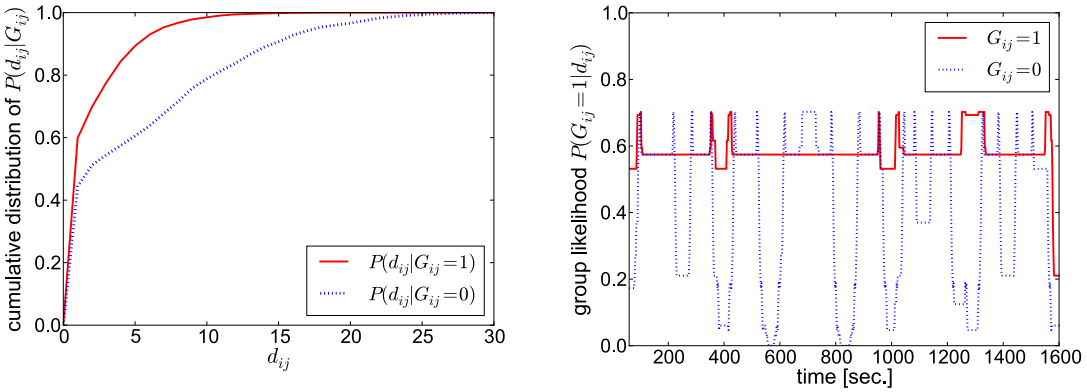
5.5.1 Constructing Group Classifier

At first, we modeled proximity and trace-similarity of the groups based on the learning dataset, and then synthesized these models to construct a group classifier.

Proximity Model

Analyzing the learning dataset, we calculated the distribution of n_{ij}, which has been introduced in Eq. (5.1) as a proximity measure between two clients. Fig. 5.12 (a) shows the distribution in two different cases: One is for the pairs in the same group ($G_{ij} = 1$) and another is for pairs in different groups ($G_{ij} = 0$). As shown in the resulting distribution, n_{ij} is not greater than 1 in more than 90% cases for the pairs in different groups, while n_{ij} is not less than 2 for as much as 95% of the same group cases. Thus, we can accurately distinguish whether or not a pair of clients belongs to the same group by observing the recent RSS measurements.

To examine the classification capability of the proximity feature, we tried constructing a group classifier using only the proximity model. We picked out RSS logs of three clients, say A_i, A_j, and $A_{j'}$ from the test dataset, where A_i and A_j belong to the same group while $A_{j'}$ is not. After applying the proximity-based group classifier to the pairs (A_i, A_j) and $(A_i, A_{j'})$, we obtained a series of group likelihood. The result is shown in Fig. 5.12 (b). For



(a) distribution of edit distance

(b) group likelihood (edit distance)

Figure 5.13: Trace-based group classification model

the pair (A_i, A_j), the group likelihood is successfully around 1 throughout the experiment. As for $(A_i, A_{j'})$, the likelihood is less than 0.1 almost throughout the experiment. A problem is that the group likelihood of $(A_i, A_{j'})$ rises around $t = 1,100$ seconds and $t = 1,500$ seconds. This is because these two groups were going around nearby booths at that time. The trace-similarity model, which will be discussed in the next section, helps to distinguish such nearby groups.

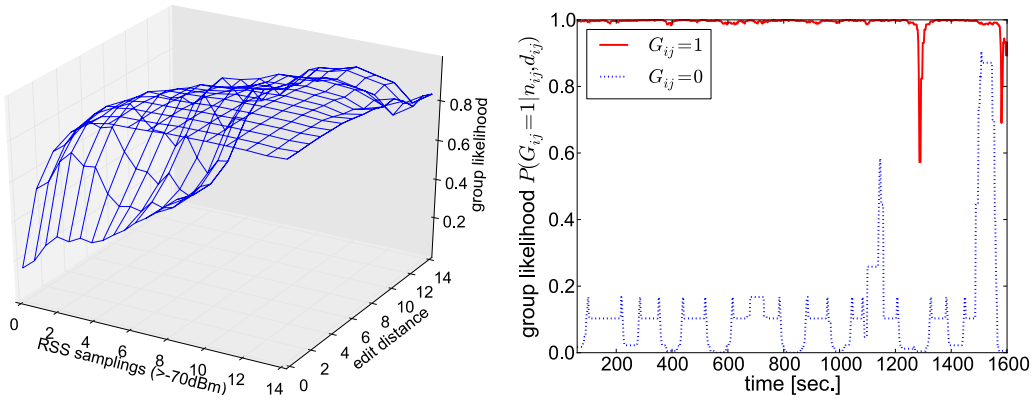
Trace-similarity Model

We calculated the distribution of d_{ij}, which has been defined as a trace-similarity measure in Eq. (5.2). The resulting distribution is shown in Fig. 5.13 (a). Since we conducted this experiment in an exhibition, users spent much time staying at each booth. This leads the distribution of d_{ij} to be biased to zero. However, the probability that $d_{ij} \geq 5$ is around 40% for the clients in the same group while the corresponding probability for the different group cases is no more than 10%. This difference takes an important role in distinguishing nearby groups.

We also tried constructing a group classifier using only the trace-similarity model, and then applied the resulting classifier to the traces of pairs (A_i, A_j) and $(A_i, A_{j'})$, which correspond to those in the previous section. As a result, we obtained a series of group likelihood shown in Fig. 5.13 (b). When the two clients are both staying at a booth, the group likelihood is relatively high regardless of whether they are in the same group or not. On the other hand, while the clients travel between the booths, the likelihood in different group cases falls to around zero. The latter feature contributes to separation of nearby groups, as mentioned above.

Group Classifier

Finally, we synthesize these two models using the Bayes' rule in Eq. (5.7) to construct a complete group classifier. The resulting group estimation model is shown in Fig. 5.14 (a).



(a) group estimation model (b) group likelihood (synthesized)

Figure 5.14: Group estimation model

Fig. 5.14 (b) shows the group likelihood for pairs (A_i, A_j) and $(A_i, A_{j'})$ based on the resulting group classifier. Recalling the proximity-based group likelihood in Fig. 5.12 (b), the likelihood of $(A_i, A_{j'})$ unsuccessfully rises when their groups are nearby. In Fig. 5.14 (b), the failure around $t = 1,100$ seconds is mitigated since the large trace dissimilarity suppressed the group likelihood. Thus the proximity and trace-similarity models complement each other to achieve better grouping accuracy.

5.5.2 System Performance

We evaluated the performance of PCN in terms of grouping accuracy and relative position error.

Grouping Accuracy

Appropriate group estimation is a key to enhance positioning performance with our context-supported error correction mechanism. We applied the group classifier to all the sensor data and RSS logs in the test dataset to extract groups for each client. Then we evaluated the grouping accuracy by the accuracy rate of *pairwise membership test*: For each pair of clients, we checked whether they are in the same group or not, and compared them with the actual grouping. As shown in temporal change of the resulting grouping accuracy in Fig. 5.15, we successfully achieved accuracy ratio of more than 90% throughout the experiment, with average grouping accuracy of 98%.

Relative Position Error

To evaluate the efficacy of our context-supported error correction mechanism, we evaluated errors in relative positions to the nearby clients that are within 10m from each client A_i of interest. In order to evaluate relative position errors, we need to collect accurate ground truth traces of the users. However, it is difficult to continuously track correct location of each user in such an indoor event place where GPS is not available. Therefore we

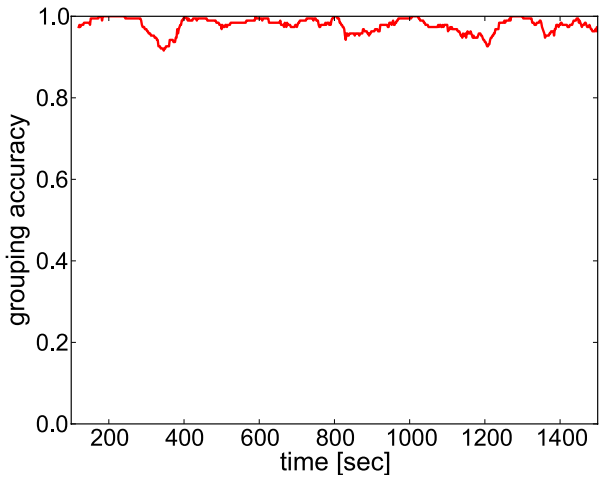


Figure 5.15: Accuracy of group estimation

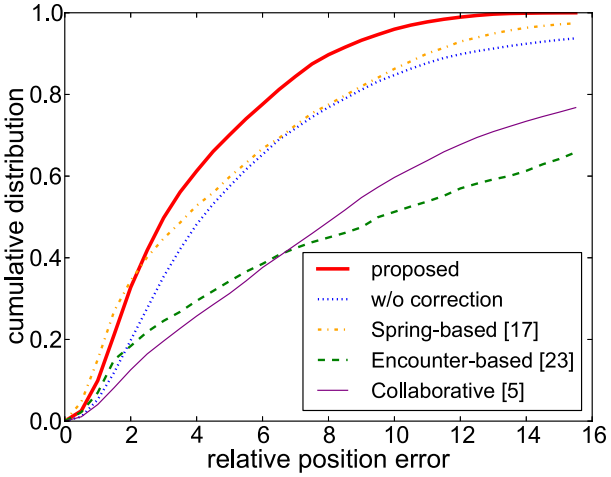


Figure 5.16: Relative position errors (field experiment)

recorded coordinates of the users with corresponding timestamps when they arrive at and leave booths, and then evaluated relative position accuracy only while the users are stopping at a booth. To complement the limitation in the field experiment, we have also conducted simulations assuming the trade fair, which will be described in Section 5.6.

We applied our context-supported relative positioning algorithm (referred to as *Proposed* hereafter) to the test dataset, and evaluated the relative position error every 2 seconds. In Fig. 5.16, we compare the cumulative distribution of the relative position errors with the following four approaches:

Without-correction: Basically the same algorithm as the *Proposed*. The only difference is that it does not apply group-based error correction in the trace and distance estimation processes. Comparison with this method clarifies the effectiveness of our

heuristic error correction mechanism.

Spring-based: The method proposed in our previous work [97], which also employs group-based error correction. Essentially, difference from *Proposed* is two-fold: (i) After deriving group members of each client, *Spring-based* merges the estimated groups based on the following transitivity assumption: Consider three clients A_1, A_2 and A_3. If client pairs (A_1, A_2) and (A_2, A_3) have group relationship, respectively, then we regard that the pair (A_1, A_3) also have group relationship regardless of the group likelihood between A_1 and A_3. This merge process guarantees consistency of the group estimation results among the clients and reduces the false negative group relationships at the prices of some additional false positives. As we will analyze in Section 5.6.7, the false positive group relationships have relatively large impact on the trace and position errors since it causes inappropriate trace/distance corrections. *Proposed* does not perform this merge process to avoid the additional false positives in the group estimation. (ii) In the positioning phase, *Spring-based* generates virtual attracting force between the clients according to the Bluetooth RSS values that have been observed during the current time step. Then the relative positions between the clients are iteratively refined by correcting their position according to the aggregate force applied to each client. Based on the proximity assumption between the group members, it heuristically adds weak attracting force between the clients in the same group, while applying weak repulsive force between those in different groups.

Encounter-based: Collaborative PDR proposed in [38]. For this method, we assume that the initial positions of the clients are known by, *e.g.*, manual input by the users. Then it periodically updates their estimated positions using the original movement vectors obtained by PDR. Whenever surrounding clients are detected by a Bluetooth scan, their estimated positions are corrected to alleviate accumulated errors. Weighting the original position estimates of each client with their confidence levels, the weighted average value is adopted as their common new position estimates.

Collaborative: Basically the same algorithm as the *Encounter-based*. A difference is that it employs the position correction algorithm in the collaborative WiFi-based localization [69].

For *Encounter-based*, large position errors continuously occur, resulting in the median localization error of more than 10m. In this approach, relative position estimates between the clients are reset to $(0, 0)$ whenever they come close within the radio range of Bluetooth (typically 10-15m). Although such encounter-based correction effectively bounds PDR errors of individual users in large-scale outdoor environments, the required position granularity in indoor environments (*e.g.*, exhibitions) is much smaller than the radio range. It may rather increase the relative position errors unless distance to the neighbor is sufficiently small.

Collaborative mitigates the problem in *Encounter-based* by utilizing the RSS-based distance information. Instead of simply adjusting the estimated positions of the neighboring clients to exactly the same location, it corrects the position estimates such that they become consistent with the estimated distance between the clients. This helps to avoid the additional position errors caused by the encounter-based correction. However, it may often happen that Bluetooth radio signals between the clients are attenuated by human bodies or walls, which makes the estimated distance to be larger than actual values. Consequently, there still remain large position errors with a median localization error of 8.20m.

Without-correction reduces relative position errors compared to both *Encounter-based* and *Collaborative* by carefully correcting relative distance between the clients. An essential difference from the *Collaborative* is that it corrects relative positions between the clients only when distance between their current position estimates is larger than the estimated distance based on the RSS information. This effectively avoids inappropriate position adjustment due to the signal attenuation by obstacles, which was frequently observed in our field experiment. In combination with the position update algorithm based on the relative position graphs, it achieves a median localization error of 4.16m.

Spring-based and *Proposed* further pursue better relative position accuracy by heuristically correcting trace/distance errors based on the group relationship between the clients, and finally achieved median relative positioning accuracy of 3.71m and 3.01m, respectively. As mentioned above, the group merge process of *Spring-based* increases the false positive group relationships, which incur inappropriate trace/distance correction. Actually, group estimation accuracy of *Spring-based* declines to 90% (9.9% false positives and 0.1% false negatives) after the merge process. In terms of enhancing the relative position accuracy, it would be better to minimize the false positive group relationships rather than keeping the consistency of group estimation results among the clients. In addition, improvement of the positioning algorithm would also contribute to enhance robustness of the relative position estimation. Although group-based distance correction mechanism of *Spring-based* helps to enhance positioning accuracy to some extent, it does not guarantee how much the peer-to-peer distance is corrected. On the other hand, *Proposed* corrects estimated distance in a more systematic manner based on the RSS model obtained by the preliminary measurement experiment, as described in Section 5.3.4. Also, *Proposed* introduces a relative position graph for the positioning phase, instead of the dynamics-based position adjustment. By considering confidence of the current estimated positions of each client in determining the final relative position estimates, it tries to minimize the impact of accumulated PDR errors on the relative position accuracy.

5.6 Simulation

Performance of PCN would depend on the mobility characteristics of the users. To complement the evaluations in Section 5.5, we have also conducted extensive simulations and

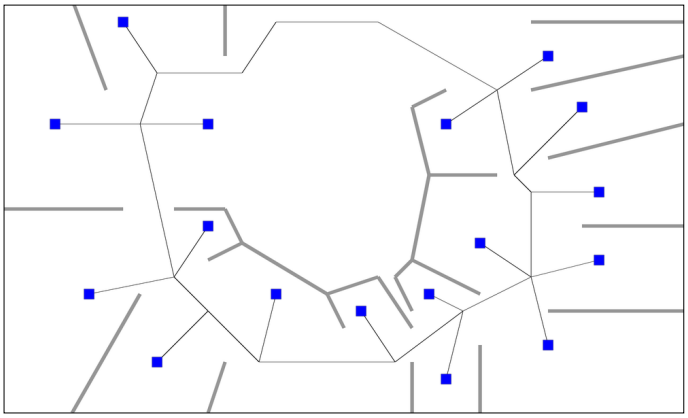


Figure 5.17: Field map for simulation experiments

analyzed the performance of PCN under various scenarios.

5.6.1 Assumptions and General Simulation Settings

In the simulations in the following sections, we assume a 40m×27m field that models the exhibition hall in Fig. 5.10. Totally 17 points of interest (POIs) are placed at each booth and passages are modeled by the 31 line segments connecting the POIs and additional waypoints. Fig. 5.17 shows a snapshot from our simulator where the POIs and passages are indicated by square markers and solid lines, respectively. Thick lines separating the booths represent walls, which will be considered in Section 5.6.5. To generate user mobility, we employ the following mobility model that simulates people's collective activity. Let N_g be the number of groups. As with Reference Point Group Mobility Model [98], we generate N_g reference points each of which defines average behavior of a group. Each reference point randomly selects a destination POI and then moves toward the selected destination along the shortest path on the passages. The moving speed is also chosen randomly from (v_{min}, v_{max}), where v_{min} and v_{max} are the minimum and maximum speed of the clients, respectively. Unless otherwise noted, we assume $v_{min} = 0.5$ m/s, $v_{max} = 1.5$ m/s. After arriving at the destination POI, the reference point stays at the current position for random duration between 0 and 100 seconds and then leaves toward the next POI. Whenever a reference point of a group decides the next destination waypoint, corresponding group members also select their own destination from the region of 1.5m from the reference point. The moving speed of the clients is decided so that they arrive at the destination waypoint at the same time as the reference point. Thus, clients in the same group move similarly to form collective activity. The mobility model also allows groups to change dynamically. For this purpose, we specify typical size of the groups and probability of group change, denoted by N_s and p_g, respectively. At each time step (with duration of τ seconds), each client leaves its current group with probability p_g. Given p_g and τ, the expected duration that a client stays in a group is τ/p_g seconds. When leaving the group, the client selects the next group to join according to the current size of each group; the

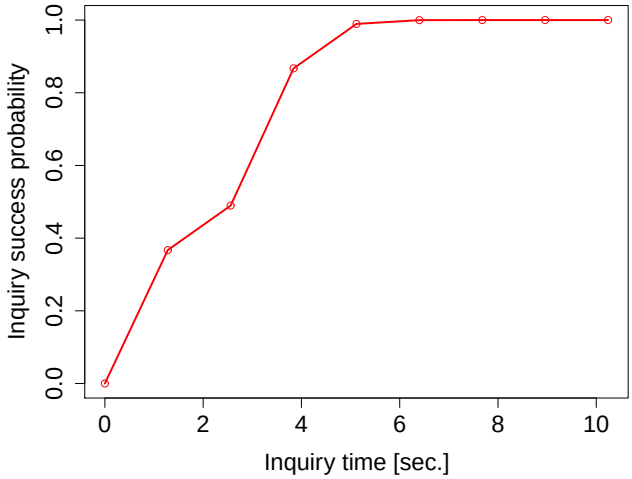


Figure 5.18: Percentage of devices expected to be discovered in a Bluetooth scan

groups with fewer members are more likely to be chosen as the destination. Thus, we can control the size of the groups by the parameter N_s, allowing small variance. In the simulations, we set the number of clients to $N_g N_s$. By default, we assume $p_g = 0$, $N_g = 5$ and $N_s = 4$.

Given the ground truth traces, we break the trace of each user into segments of 2 seconds long to obtain a sequence of correct movement vectors. Then, for each movement vector, we further divide it into step vectors of 0.7m long, assuming typical step length of the users. If there remains a fraction below 0.7m, it is distributed over all the step vectors evenly. Finally, we add noise to each step vector based on the step-level PDR error model, which has been defined in Section 5.3.3. By reassembling these step vectors, we generate a noisy movement vector that is expected to be obtained by PDR. In addition, we also consider false positive detection of user steps due to unexpected motion of the client device. We model occurrence of such wrong detection by a Poisson process with mean interval of 20 seconds and simulate it by adding three step vectors with a length of 0.7m, each of which has random orientation.

To generate RSS logs, we model the communication via Bluetooth as follows: Each client has a state, which is either of *scanning* or *discoverable*, and periodically alternate its state based on the scan control mechanism in Section 5.3.1. Clients in a *scanning* state perform a Bluetooth scan with duration of 5.12 seconds and repeatedly broadcast inquiry messages. Then neighboring devices that are not performing a scan at the same time (*i.e.*, in a *discoverable* state) probabilistically reply a response message so that the scanning device can collect Bluetooth RSS. Note that the neighbors in the *scanning* state can hardly respond to the inquiries from other devices since they quickly change the radio frequency during the scan process. Thus the detection probability of each neighbor depends on the duration of time when the neighbor is in the *discoverable* state during the scan process of the device itself (*i.e.*, effective scan duration). Assuming a perfect commu-

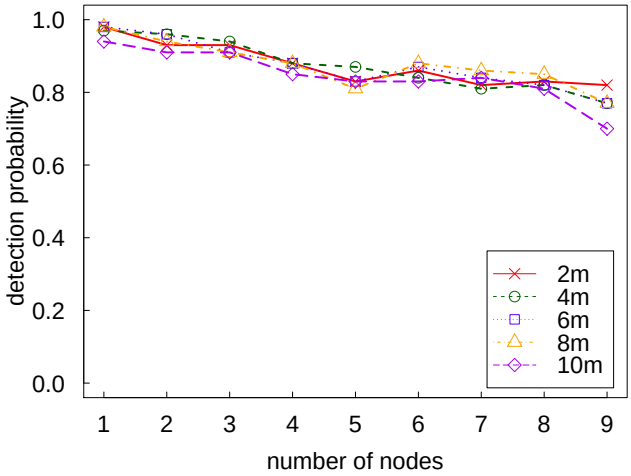


Figure 5.19: Impact of client density on neighbor detection ratio

nication channel, detection probability of a neighbor reaches 99% when the neighbor stays in the *discoverable* state during the entire scan process [99]. The detection probability with other effective scan duration is plotted in Fig. 5.18. In the simulations, we determine the detection probability of each neighbor based on the theoretical model in Fig. 5.18 and the effective scan duration with respect to each neighbor. In addition, we also consider the degradation of neighbor detection probability due to collision of inquiry messages. To model the Bluetooth interference from the neighboring clients, we have conducted the following experiment using Nexus S phones: Five phones (called *targets*) were arranged in line with equal spacing of 2m, listening to the inquiries from neighboring devices. Then other phones (referred to as *scanners*) are placed at the same location and repeatedly perform Bluetooth scans to collect RSS from those five targets. Each scanner probabilistically determines its timing of Bluetooth scans based on the scan control mechanism in Section 5.3.1. Varying the number of scanners between 1 and 9, we collected more than 200 Bluetooth scan logs for each case. Fig. 5.19 shows the detection probability of each target located at different distances. Regardless of distance to the targets, the detection probability linearly declines as the number of scanners increases. Based on the result above, we decrease the neighbor detection probability in Fig. 5.18 by $0.02N_n$, where N_n is the number of clients within the wireless communication range of the client performing a scan. The Bluetooth RSS from a neighboring client at distance d is assumed to follow a Gaussian distribution $\mathcal{N}(\mu(d), \sigma^2(d))$, where $\mu(d)$ and $\sigma(d)$ are the mean and standard deviation of the RSS values observed at distance d in our preliminary experiment in Section 5.2.2. We have also seen in Fig. 5.9 that signal attenuation by human bodies significantly affects Bluetooth RSS; once the signal propagation path is obstructed by any pedestrian, the observed RSS value poorly contributes to group estimation and position refinement since it rarely exceeds the threshold of -70dBm. In the simulations, we model the user body by a line segment with a length of 0.248m, which is a statistical average of the width

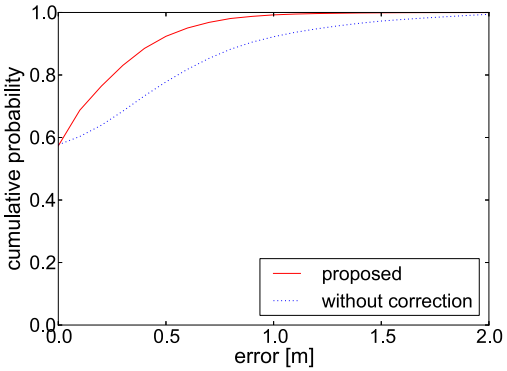


Figure 5.20: Errors in movement vectors

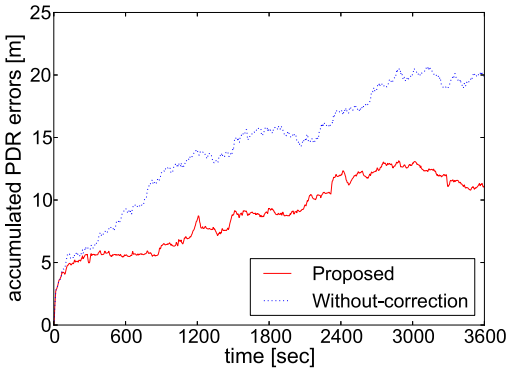


Figure 5.21: Accumulated PDR errors

of human waist, and assume that Bluetooth signals are blocked by the human bodies. The user body is placed at a distance of d_u from the current location of the client device, where d_u depends on the current placement of the device. We assume that $d_u = 0.30$m when the device is held in hand and $d_u = 0.01$m when it is put in the pocket.

With the settings above, we can simulate collective activity of people in a controllable manner and generate PDR traces and RSS logs. In the following sections, we will show the simulation results with various scenarios to clarify efficacy of the PCN system. The simulation time is set to 3,600 seconds for each experiment. Unless otherwise noted, we assume that all the users hold their client device in hand.

5.6.2 Effectiveness of Group-based Error Correction

Due to limitation of the ground truth collection in the field experiment, in Section 5.5.2, relative position errors were evaluated only when the users were stopping at a booth. To complement the insufficiency of the performance evaluation, we first examine the accuracy of trace and relative position estimation over the whole simulation time. Fig. 5.20 shows the cumulative distribution of errors in movement vectors. We can see that 60% of the movement vectors are not affected by any PDR errors since the noise can arise only when the clients move between the booths, or false positive step detection due to unexpected device motion occurs. Our goal is to refine the accuracy of the remaining 40% movement vectors that contain some errors. The result shows that 23% of the original movement vectors contain errors of more than 0.5m, while the corresponding ratio is reduced to less than 8% after the group-based trace correction. Considering the traces when the clients are moving between the booths, average errors in movement vectors are mitigated from 0.66m to 0.40m, which corresponds to an improvement of 39%. Note that the errors in movement vectors are accumulated in the PDR traces every time step of 2 seconds. Fig. 5.21 shows the accumulated PDR errors at each time step, averaged among all the clients. The trace error gradually increases with time and finally becomes more than 20m. The group-based trace correction effectively mitigates the error growth and achieves an average accumulated trace error of 11.1m after the whole simulation time of 3,600 seconds.

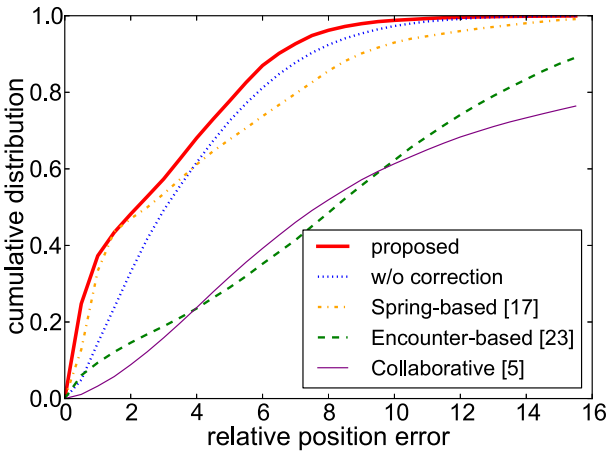


Figure 5.22: Relative position errors (simulation)

Finally, we show the cumulative distribution of relative position errors in Fig. 5.22. The trend of the errors is similar with those we have observed in the field experiment. By the group-based error correction mechanism, the median positioning error is reduced from 3.1m to 2.2m, achieving an improvement of 29%.

5.6.3 Performance with Different Phone Poses

In practical situations, people may not always hold their phone in hand. In order to examine the impact of device placement on grouping and trace estimation accuracy, we compare the performance of PCN in three scenarios where (i) all the users hold their phone in hand, (ii) all the users put their phone in a trouser pocket and (iii) for each group of four persons, two users hold their device in hand and the others carry in their trouser pocket.

For trace estimation with the *in-pocket* case, we assume that initial offset of the device heading with respect to user heading is known. As mentioned in Section 5.4, the direction offset estimation can be done by tracking the pose of the device using gyro sensors. Given the offset value, we can also apply the PDR algorithm in Section 5.2 for the *in-pocket* case. In the simulations, we assume that PDR errors with the *in-pocket* devices follow the pose-dependent model constructed in Section 5.4, where the variance of heading direction errors increases by 23^2 degrees compared to the *in-hand* case. We also assume that the current pose of the device can be estimated by each client and reported to the server every time step.

Simulation results with each scenario are listed in Table 5.1. When the device is carried in pocket, Bluetooth signals from neighboring clients are more likely to be attenuated by the user body, reducing the opportunities to correct relative distance between the clients. In addition, larger PDR errors of the *in-pocket* devices incur more rapid error accumulation in the estimated positions. Consequently, absolute errors in movement vectors and relative positions gradually increase with the ratio of users who carry their device in pocket.

Table 5.1: Impact of phone poses

	Scenario (i)	**Scenario (ii)**	**Scenario (iii)**
grouping accuracy	98.0%	94.1%	96.3%
errors in movement vectors (*Proposed*, *w/o-correction*)	0.40m / 0.66m	0.52m / 0.75m	0.45m / 0.71m
relative position errors (*Proposed*, *w/o-correction*)	2.20m / 3.06m	2.67m / 3.93m	2.37m / 3.58m

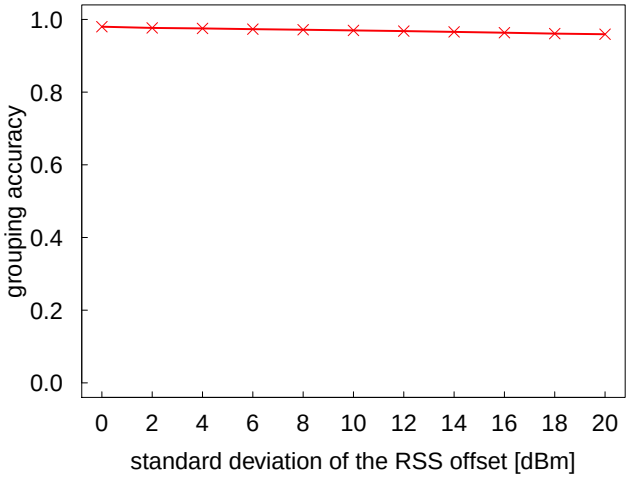


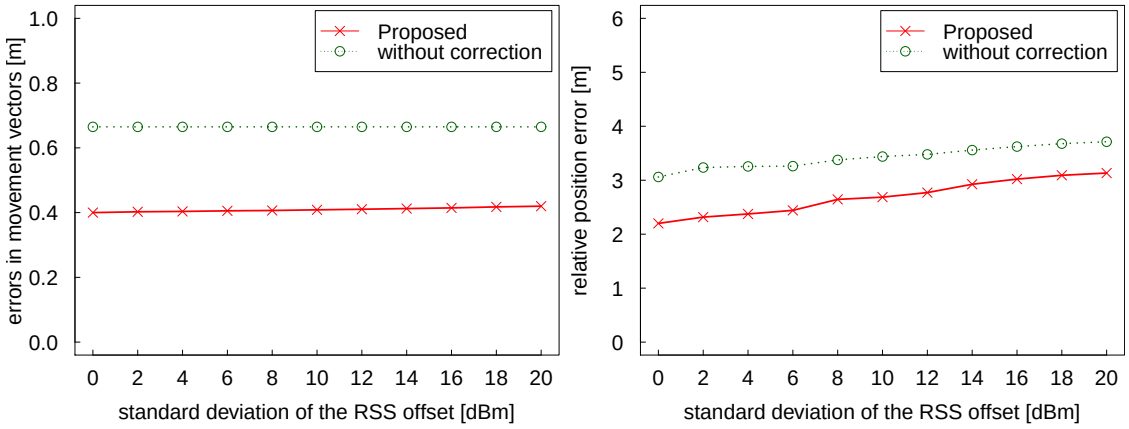
Figure 5.23: Grouping accuracy in heterogeneous environment

Nevertheless, PCN still maintains the high grouping accuracy of 94.1% even if all the users carry their phone in pocket (*i.e.*, Scenario (ii)). Accordingly, average errors in both movement vectors and relative positions are effectively reduced by approximately 30% in all the scenarios.

5.6.4 Robustness to Client Heterogeneity

Since signal transmission power and reception sensitivity of Bluetooth modules may be different among individual phone models, it often happens that characteristics of Bluetooth RSS may vary for each pair of clients. Through the following simulations, we examine robustness of the PCN system against such heterogeneity of client hardware. Here, we assume that RSS of the Bluetooth beacons from a client A_i to another client A_j follows a Gaussian distribution $\mathcal{N}(\mu(d) + \Delta_{ij}, \sigma^2(d))$. Note that $\mu(d)$ and $\sigma(d)$ are the mean and standard deviation of the RSS values that are observed at distance d in our preliminary experiment using Nexus S phones (see Section 5.2.2). By adding a random offset value Δ_{ij}, we simulate the difference in signal transmission characteristics of the clients. We assign a different offset value for each pair of clients, and assume that it follows a Gaussian distribution $\mathcal{N}(0, \sigma_\delta^2)$. Other simulation settings are the same as the ones we have described in Section 5.6.1. Varying the standard deviation σ_δ from 0 dBm to 20 dBm, we have conducted simulations of 3,600 seconds and evaluated performance of the PCN system.

Fig. 5.23 shows average grouping accuracy with each σ_δ. We can see that PCN keeps



(a) errors in movement vectors (b) relative position error

Figure 5.24: Error correction capability in heterogeneous environment

high grouping accuracy even if the characteristics of Bluetooth RSS can significantly vary among the client pairs; it still achieves the grouping accuracy of 96% with $\sigma_\delta = 20$ dBm. Basically, the radio signals are attenuated in proportion to $1/d^2$, where d is the distance from the transmitter. Therefore, even with presence of the hardware heterogeneity, large RSS values are still likely to be observed between the clients that have been in close proximity to each other for a long period of time. In addition, PCN combines two feature values that are extracted from independent sources (*i.e.*, Bluetooth RSS and estimated traces by PDR) to enhance robustness of the group estimation. While the client heterogeneity may degrade reliability of the proximity feature, the trace-similarity information would contribute to correct the group likelihood and thus prevents potential group estimation errors.

Owing to the robust group estimation, the group-based error correction mechanism of PCN is also effective in the heterogeneous environment. Fig. 5.24 shows average errors in movement vectors and relative position errors with each σ_δ. Since PCN keeps high grouping accuracy regardless of σ_δ, the impact of heterogeneity on trace estimation accuracy is also substantially small. While the relative position errors gradually increase with σ_δ due to the hardware-dependent offset in RSS values which directly affects RSS-based distance estimation, the group-based error correction still effectively improves the position accuracy by 16-28%. Thus PCN has sufficient robustness against client heterogeneity.

5.6.5 Impact of Walls

Bluetooth radio signals can be attenuated not only by human bodies but also walls in the environment. In this section, we evaluate the performance of PCN in a scenario that contains walls. We separate adjacent booths by a wall and assume that the walls block the radio signals. The locations of the walls are indicated by thick lines in the floor plan in Fig. 5.17. Except for the presence of the walls, simulation configuration is basically the same as that of Section 5.6.1.

Table 5.2: Impact of walls

	w/o walls	**w/ walls**
grouping accuracy	98.0%	98.2%
errors in movement vectors (*Proposed*, *w/o-correction*)	0.40m / 0.66m	0.40m / 0.66m
relative position errors (*Proposed*, *w/o correction*)	2.20m / 3.06m	2.45m / 3.21m

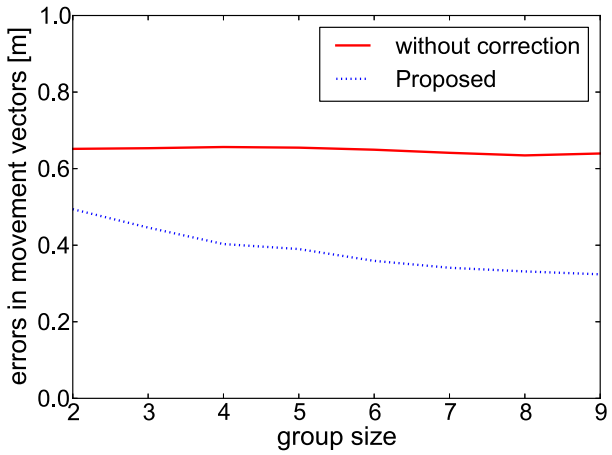


Figure 5.25: Impact of group size on trace estimation error

Table 5.2 outlines the simulation results with/without walls. The relative position errors increased by 10% with the presence of walls since Bluetooth communication between the clients in adjacent booths is likely to fail, reducing the opportunities to correct relative distance between the clients. On the other hand, the signal obstruction by the walls tends to positively impact the grouping accuracy since it contributes to discriminate the groups that are staying at adjacent booths by suppressing their proximity feature.

5.6.6 Impact of Group Size

Size of the groups is an important factor that can affect the positioning performance. As the group size gets larger, trace errors due to random sensor noise are expected to be alleviated. Furthermore, clients that temporarily exhibit different behavior from their group members can be detected more robustly by the outlier detection in Section 5.3.3, reducing the average trace estimation error by preventing such deviated traces from affecting the trace estimation of other group members.

To confirm the assumption, we ran simulations with various group sizes. Fig. 5.25 shows the average errors in movement vectors where the number of clients in a group is varied from 2 to 9. As with the simulations in the previous sections, the number of groups is set to $N_g = 5$. Obviously, average errors in original movement vectors are independent of the group size since the PDR traces are calculated individually by each client. As we expected, trace estimation errors were more effectively reduced as the group size gets larger. In case that $N_s = 9$, the average error in movement vectors was reduced from 0.64m to 0.32m, which corresponds to 50% improvement. More interestingly, average

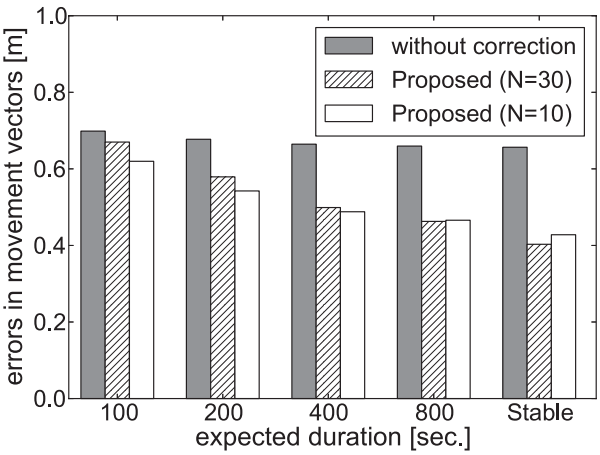


Figure 5.26: Errors in movement vectors with different group change frequencies

trace error was alleviated by 24% even if the group size was only two. Thus the benefit of group-based trace error correction can be fully derived when people form large groups, while we can also effectively reduce trace estimation errors with small groups consisting of a few people. While the neighbor detection ratio in Bluetooth scans gradually declines as the density of the clients increases, PCN keeps a grouping accuracy of 96.4% even if the group size $N_s = 9$, which is only 1.6% lower than the case with $N_s = 2$.

5.6.7 Flexibility to Group Change

To evaluate flexibility to people's movement between different groups (*i.e.*, group change), we conducted simulations varying the group change probability p_g. In this experiment, we also tried a shorter window size of $N = 10$ in the group estimation, in addition to the evaluations with $N = 30$. Fig. 5.26 and Fig. 5.27 represent average errors in movement vectors and relative positions, respectively, where $p_g = 0.02$, 0.01, 0.005, 0.0025 and 0. Since the length of each time step τ is 2 seconds, expected duration of each client to stay in a group ranges from 100 seconds ($p_g = 0.02$) to 800 seconds ($p_g = 0.0025$). In case that $p_g = 0$, all the clients stay in their original groups throughout the simulation.

As seen in Fig. 5.26, errors in movement vectors are reduced more effectively as the group change probability is low. It is natural since the clients can benefit from the trace error correction only when they move together with some others; for the clients moving between the groups alone, basically, original PDR trace is directly adopted to the final trace estimate. When the groups are entirely stable (*i.e.*, $p_g = 0$), PCN effectively reduced the errors in movement vectors by 39% with $N = 30$. Similar characteristics are also seen in the positioning accuracy in Fig. 5.27. PCN achieved up to 28% improvement in the relative position accuracy.

On the other hand, in case that the frequency of group change is extremely high, group-based error correction may rather incur additional errors in the estimated positions, as seen at $p_g = 0.02$ (100 seconds) in Fig. 5.27. The errors are mainly caused by the transient

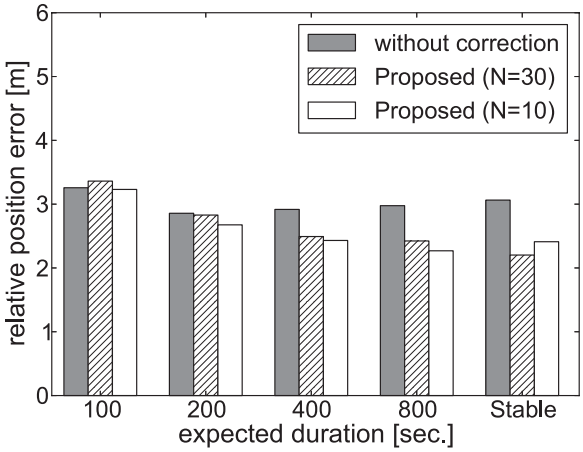


Figure 5.27: Relative position errors with different group change frequencies

failure in group estimation that may occur when a user is leaving a group. As mentioned in Section 5.3.2, PCN estimates group relationship based on all the movement vectors and RSS logs, which have been collected during the recent $N\tau$ seconds. While it is necessary for enhancing robustness of group estimation allowing errors in movement vectors and estimated distance, it also causes delay for the PCN system to detect the users who have started different behavior from the group members (*i.e.*, leaving a group). Since inappropriate correction would be applied based on wrong group information until the group likelihood falls below the threshold Θ_{group}, relative position accuracy may be temporarily degraded at the time of the group change. A possible solution for this problem is to adopt smaller window size in the group estimation. In Fig. 5.26 and Fig. 5.27, we can see that the errors in estimated traces and relative positions at $p_g = 0.02$ are both effectively mitigated when the window size is $N = 10$. Since the flexibility to the group change trades off with the robustness of group estimation, we should carefully select the window size according to expected characteristics of user mobility such as frequency of group change and variance of trajectories within a group. For example, in the situations where groups are assumed to change frequently (*e.g.*, in a party place), smaller N would provide better performance by avoiding the wrong correction. In the exhibition where people tend to form stable groups, larger N will enhance the robustness of group estimation. Note that positioning performance of PCN converges to that of a simple PDR/encounter-based relative positioning if people do not form any groups and move independently.

5.7 Discussion

5.7.1 Utilizing Group Information for People-centric Navigation

Advantage of detecting collective activity is not only accuracy improvement but also helping human perception of location. For mobile social navigation, contextual information representing behavior of surrounding crowd like “five people are standing together on your

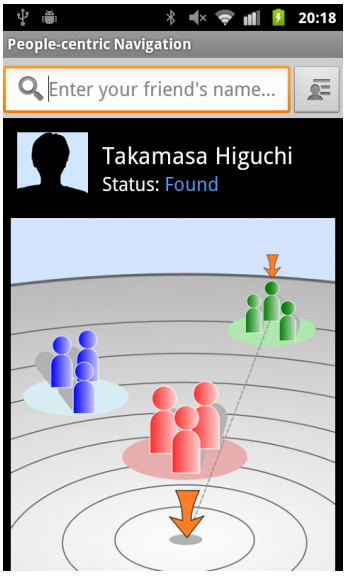


Figure 5.28: PCN client (prototype)

left" or "a large pedestrian flow is crossing in front of you" may enable them to serve as instant *landmarks* for the user to recognize position relationship to the surrounding people. In navigation, directions like "follow the pedestrian flow you are belonging to, so you can get to the destination" may provide an informative hint for guiding the user in crowded situations. Fig. 5.28 shows a screenshot from our PCN client on the Android platform. Based on the group information, the user can identify the target person in the group of three people behind another group of three people crossing in front of him. Thus, by fully utilizing the advantage of collaboration, PCN heuristically corrects potential errors in estimated positions and associates them with collective activity context to totally achieve "people-centric" navigation as its name suggests.

5.7.2 Challenges

Although this chapter has mainly focused on crowd mapping in a local space like a party place or an exhibition hall, we believe that the core idea of PCN can be also applied to more general, public situations. In crowded situations like a shopping mall or a busy station, people moving toward the same direction naturally form pedestrian flows. As with the "groups" we have considered in this chapter, people in a pedestrian flow are expected to have similarity in their activity, while the variety in timing of movements and shape of the traces between individuals may become larger than the small groups. Such similarity could also be utilized to improve positioning accuracy and human perception of position relationship.

On the other hand, there still remain some potential issues to be addressed in applying the context-supported positioning to more general cases. Firstly, we cannot always assume that most of the people in the crowd participate in the service, offering their sensor data and Bluetooth RSS logs. In terms of enhancing applicability to various situations, it must

be preferable that we can recognize the surrounding situation even if sensor data can be collected from only a small proportion of people. To recognize groups with the limited sensor data, we may partially employ the idea of crowd density estimation and pedestrian flow detection techniques, which have been mainly studied in the area of computer vision [100]. Some recent work like [101, 102] presented Bluetooth-based collaborative crowd density estimation using mobile phones. By sampling the groups' behavior from the collected sensor data, and then detecting size of the groups using crowd density estimation, we could offer a rough crowd map, which supports people to know the surrounding situation even if the ratio of participants is low.

Energy efficiency is also an important factor that motivates users to participate in the collaborative sensing. The current version of PCN assumes that each client sends sensor data and RSS logs to the centralized server every 2 seconds to provide a local map in a timely fashion. Although adopting a longer upload interval could save energy consumption for wireless communication, it also increases the delay until the local map is updated. The collective activity context may also offer a reasonable solution to the dilemma; once groups are detected using the past sensing logs, incoming behavior of a group member can be predicted from that of other group members based on the assumption of trace similarity and proximity properties. Therefore, the crowd map can be updated even if all the clients in a group do not send the sensing results at every time step. Thus the clients can alleviate the data upload frequency as far as the sensing results can be collected from a sufficient proportion of people in each group.

5.8 Conclusion

In this chapter we have presented PCN, a novel positioning system that provides a local map of surrounding crowd. PCN estimates relative position to the surrounding people based on sensor readings and Bluetooth RSS, both of which can be easily obtained via off-the-shelf mobile phones. Utilizing the feature of "group activity," it reduces the effect of sensor noise and other error-inducing factors. Through a field experiment in a real trade fair, we demonstrated that PCN improves positioning accuracy by 28% compared to a conventional approach owing to its context-supported error correction mechanisms. Furthermore, we analyzed the performance of PCN through extensive simulations and clarified the situations where the collective activity context effectively contributes to enhance positioning performance.

Chapter 6

Conclusion

This dissertation has presented a novel approach to low-cost and accurate indoor positioning of mobile devices. The goal of this dissertation is to cope with a common problem in existing indoor positioning systems; the trade-off between accuracy and their costs in terms of infrastructure deployment and calibration effort. We have addressed the issue by employing cooperative approaches where neighboring mobile devices collaborate with each other to achieve better positioning performance. To maximize effectiveness of the collaboration, our mobility-aware cooperative localization incorporates observations on common characteristics of human mobility (*e.g.*, stop-and-go behavior and collective activity) into the algorithm design of localization systems. In this dissertation, we have made the following three primary contributions to embody this idea.

Firstly, we have presented a fully-decentralized algorithm to collaboratively localize mobile nodes with a small number of anchor nodes and a reduced amount of localization attempts. We assume that at least three anchors are deployed at known locations in the target area, and all the nodes have ad hoc communication and range measurement faculties to enable accurate peer-to-peer distance measurement. Focusing on the stop-and-go behavior of mobile nodes, the proposed method detects movement of the nodes and selects only *static* nodes as pseudo-anchors. Thus it effectively prevents error propagation from the nodes in motion. Furthermore, it automatically adjusts localization frequency according to the estimated speed of each mobile node to reduce redundant localization attempts. Experimental results have shown that the average localization error of the proposed method is around 0.2m, which is substantially lower than that of existing cooperative localization methods. We have also shown that the method could reduce localization frequency by up to 77% without degrading the tracking performance.

Secondly, we have designed a general framework for performance analysis of the mobility-aware cooperative localization algorithms that exclude the nodes in motion from the set of reference points. In order to clarify when and how they can achieve optimal performance, we have derived the theoretical lower bound of the localization errors. Through a case study, we have compared performance of the proposed localization algorithm above with the theoretical bound, and shown that the errors asymptotically approach the lower bound as more than 50% of neighboring devices maintain correct movement state. The

framework of error analysis is also applicable to any other cooperative localization algorithms that select pseudo-anchors based on their movement state. As well as the error analysis, performance of the mobility-aware approaches has been also analyzed from various aspects, seeking strategies to maximize their effectiveness. Based on extensive simulations and experiments using Android smartphones, we have shown several important observations regarding anchor deployment strategy, effectiveness of accelerometer-based motion detection and combination with PDR-based trajectory estimation, etc.

Thirdly, we have designed a novel positioning system that provides a local map of surrounding crowd. Assuming the situations where anchor deployment and accurate peer-to-peer distance measurement are not available, our goal here is to provide pedestrians' position information with reasonable accuracy using only built-in functions of commercial mobile devices (*e.g.*, smartphones). The proposed system, called PCN, estimates relative positions of the surrounding people based on sensor readings and Bluetooth RSS, both of which can be easily obtained via off-the-shelf mobile phones. Utilizing the feature of "group activity," it reduces the effect of sensor noise and other error-inducing factors. Through a field experiment using Android smartphones, we have demonstrated that the error correction mechanism successfully enhances positioning accuracy by 28%.

Through these contributions, it has been shown that the mobility-aware neighbor collaboration mechanism offers improved cost/accuracy trade-offs by effectively enhancing the quality of location sensing without relying on additional infrastructure deployment or heavy calibration effort. If anchor devices can be sparsely deployed and mobile devices are capable of accurate peer-to-peer distance measurement, the proposed method in the first contribution can achieve sub-meter positioning accuracy. Otherwise PCN in the third contribution would become an alternative solution, providing a few meters accuracy using only off-the-shelf mobile phones. By combining these two approaches according to the accuracy requirement and the cost limitations, we could strongly support a variety of people-centric applications that require accurate position information of mobile devices in a ubiquitous manner.

Although we have assumed ultrasound-based distance measurement in design and evaluation of the "stop-and-go" localization in Chapter 3, its basic idea is essentially not dependent on any specific ranging techniques; other measurement means could be also applied by slightly modifying the protocol for neighbor collaboration. Combination with audio-based ranging techniques [70, 71] would provide a promising approach to energy-efficient cooperative localization on commercial mobile phones, if their calibration effort for device-dependent tuning could be mitigated somehow. In addition, the group-based error correction mechanism of PCN is applicable not only to relative positioning but also for PDR and/or RSS-based positioning systems in general. Its advantage could be fully derived even in more public scenes (*e.g.*, stations and large shopping malls) by addressing the challenges in Section 5.7.2. Through these extensions, applicability of the proposed localization algorithms could be further enhanced to underpin a wider range of mobile applications.

While this dissertation has presented the two novel positioning systems that embody the concept of the mobility-aware cooperative localization, there could be a variety of other human behavior that potentially contributes to improve the cost/accuracy trade-off. We believe it would be worth continuing to seek such further possibilities toward more efficient indoor positioning solutions.

Acknowledgement

First of all, I would like to greatly appreciate excellent supervision of Professor Teruo Higashino. Through my research activity, he often gave me helpful advice with great insight, which always guided me to recognize hidden problems and rooms for improvement. Also, I am grateful for his encouragement and support in my student life.

Secondly, I am heartily grateful to Professor Toru Hasegawa, Professor Morito Matsuoka, Professor Masayuki Murata and Professor Takashi Watanabe for their invaluable comments and helpful suggestions for improving my dissertation.

Thirdly, I thank Associate Professor Hirozumi Yamaguchi for his warmful support and direction. He kindly spared a lot of time for discussion, and always listened carefully to my opinion to consider any possibilities to refine the research work. It was precious opportunity for me to consider a matter in various aspects, and it greatly motivated me for research activity.

Also, I thank Associate Professor Takaaki Umedu from Shiga University, Assistant Professor Akihito Hiromori, and Assistant Professor Akira Uchiyama for their encouragement and support. They always gave me insightful feedbacks, and encouraged me to enjoy research work.

I also acknowledge kindful support of Dr. Sae Fujii, who graduated in 2011. As well as giving me valuable hints for research problems, she always put careful and thorough remarks on my draft for conference proceedings. I have learned a lot from her advice.

Finally, I greatly thank my family, friends and everyone in Higashino laboratory for their feedbacks, encouragement and kind support.

Bibliography

[1] Crossbow Technologies, "Micaz datasheet," 2006.

[2] K. L. Mika, M. Raento, and H. Toivonen, "Adaptive on-device location recognition," in *Proceedings of the 2nd International Conference on Pervasive Computing (Pervasive '04)*, 2004, pp. 287–304.

[3] 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," in *Proceedings of the 3rd International Conference on Pervasive Computing (Pervasive '05)*, 2005, pp. 116–133.

[4] A. Varshavsky, D. Pankratov, J. Krumm, and E. de Lara, "Calibree: Calibration-free localization using relative distance estimations," in *Proceedings of the 6th International Conference on Pervasive Computing (Pervasive '08)*, 2008, pp. 146–161.

[5] M. Klann, "Playing with fire: User-centered design of wearable computing for emergency response," in *Proceedings of the 1st International Conference on Mobile Information Technology for Emergency Response (MobileResponse '07)*, 2007, pp. 116–125.

[6] I. Constandache, R. R. Choudhury, and I. Rhee, "Towards mobile phone localization without war-driving," in *Proceedings of the 29th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '10)*, 2010, pp. 1–9.

[7] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: The design, implementation and evaluation of the CenceMe application," in *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys '08)*, 2008, pp. 337–350.

[8] Y. Wang, J. Lin, M. Annavaram, Q. A. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh, "A framework of energy efficient mobile sensing for automatic user state recognition," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, 2009, pp. 179–192.

[9] D. H. Kim, J. Hightower, R. Govindan, and D. Estrin, "Discovering semantically meaningful places from pervasive RF-beacons," in *Proceedings of the 11th International Conference on Ubiquitous Computing (UbiComp '09)*, 2009, pp. 21–30.

[10] L. Vu, Q. Do, and K. Nahrstedt, "Jyotish: A novel framework for constructing predictive model of people movement from joint WiFi/Bluetooth trace," in *Proceedings of the 9th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom '11)*, 2011, pp. 54–62.

[11] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox, and A. Schmidt, "Micro-blog: Sharing and querying content through mobile phones and social participation," in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services (MobiSys '08)*, 2008, pp. 174–186.

[12] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury, "Did you see Bob?: Human localization using mobile phones," in *Proceedings of the 16th International Conference on Mobile Computing and Networking (MobiCom '10)*, 2010, pp. 149–160.

[13] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, and M. Corner, "Virtual compass: Relative positioning to sense mobile social interactions," in *Proceedings of the 8th International Conference on Pervasive Computing (Pervasive '10)*, 2010, pp. 1–21.

[14] N. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket location-support system," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom '00)*, 2000, pp. 32–43.

[15] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, "The anatomy of a context-aware application," *Wireless Networks*, vol. 8, no. 2, pp. 187–197, 2002.

[16] D. Joho, C. Plagemann, and W. Burgard, "Modeling RFID signal strength and tag detection for localization and mapping," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA '09)*, 2009, pp. 1213–1218.

[17] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "LANDMARC: Indoor location sensing using active RFID," in *Proceedings of the 1st Annual IEEE International Conference on Pervasive Computing and Communications (PerCom '03)*, 2003, pp. 407–415.

[18] C. Wang, H. Wu, and N. F. Tzeng, "RFID-based 3-D positioning schemes," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM '07)*, 2007, pp. 1235–1243.

[19] R. J. Fontana, E. Richley, and J. Barney, "Commercialization of an ultra wideband precision asset location system," in *Proceedings of 2003 IEEE Conference on Ultra Wideband Systems and Technologies (UWBST '13)*, 2003, pp. 369–373.

[20] S. Gezici, Z. Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, and Z. Sahinoglu, "Localization via ultra-wideband radios: A look at positioning aspects for future sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 70–84, 2005.

[21] D. Jourdan, J. J. Deyst Jr., M. Win, and N. Roy, "Monte Carlo localization in dense multipath environments using UWB ranging," in *Proceedings of the 2005 IEEE International Conference on Ultra-Wideband (ICU '05)*, 2005, pp. 314–319.

[22] M. Azizyan, I. Constandache, and R. Roy Choudhury, "SurroundSense: Mobile phone localization via ambience fingerprinting," in *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking (MobiCom '09)*, 2009, pp. 261–272.

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

[24] K. K. Chintalapudi, A. P. Iyer, and V. Padmanabhan, "Indoor localization without the pain," in *Proceedings of the 16th International Conference on Mobile Computing and Networking (MobiCom '10)*, 2010, pp. 173–184.

[25] J. Yin, Q. Yang, and L. M. Ni, "Learning adaptive temporal radio maps for signal-strength-based location estimation," *IEEE Transactions on Mobile Computing*, vol. 7, no. 7, pp. 869–883, 2008.

[26] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys '05)*, 2005, pp. 205–218.

[27] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to war-drive: Unsupervised indoor localization," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys '12)*, 2012, pp. 197–210.

[28] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A reliable and accurate indoor localization method using phone inertial sensors," in *Proceedings of the 14th ACM International Conference on Ubiquitous Computing (UbiComp '12)*, 2012, pp. 421–430.

[29] M. Minami, Y. Fukuju, K. Hirasawa, S. Yokoyama, M. Mizumachi, H. Morikawa, and T. Aoyama, "Dolphin: A practical approach for implementing a fully distributed indoor ultrasonic positioning system," in *Proceedings of the 6th International Conference on Ubiquitous Computing (UbiComp '04)*, 2004, pp. 347–365.

[30] J. Albowicz, A. Chen, and L. Zhang, "Recursive position estimation in sensor networks," in *Proceedings of the 9th IEEE International Conference on Network Protocols (ICNP '01)*, 2001, pp. 35–41.

[31] C. Savarese, J. M. Rabaey, and K. Langendoen, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks," in *Proceedings of the USENIX 2002 Annual Conference*, 2002, pp. 317–327.

[32] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, 2004, pp. 50–61.

[33] J. Liu, Y. Zhang, and F. Zhao, "Robust distributed node localization with error management," in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '06)*, 2006, pp. 250–261.

[34] S. Fujii, T. Nomura, T. Umedu, H. Yamaguchi, and T. Higashino, "Real-time trajectory estimation in mobile ad hoc networks," in *Proceedings of the 12th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '09)*, 2009, pp. 163–172.

[35] J. M. Cabero, F. D. la Torre, A. Sanchez, and I. Arizaga, "Indoor people tracking based on dynamic weighted multidimensional scaling," in *Proceedings of the 10th International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '07)*, 2007, pp. 328–335.

[36] L. Hu and D. Evans, "Localization for mobile sensor networks," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking (MobiCom '04)*, 2004, pp. 45–47.

[37] S. Zhang, J. Cao, C. Li-Jun, and D. Chen, "Accurate and energy-efficient range-free localization for mobile sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 6, pp. 897–910, 2010.

[38] K. Kloch, P. Lukowicz, and C. Fischer, "Collaborative PDR localisation with mobile phones," in *Proceedings of the 15th International Symposium on Wearable Computers (ISWC '11)*, 2011, pp. 37–40.

[39] H. Wymeersch, J. Lien, and M. Win, "Cooperative localization in wireless networks," *Proceedings of the IEEE*, vol. 97, no. 2, pp. 427–450, 2009.

[40] J. Yin, Q. Yang, and L. M. Ni, "Learning adaptive temporal radio maps for signal-strength-based location estimation," *IEEE Transactions on Mobile Computing*, vol. 7, no. 7, pp. 869–883, 2008.

[41] H. Lim, L.-C. Kung, J. C. Hou, and H. Luo, "Zero-configuration, robust indoor localization: Theory and experimentation," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*, 2006, pp. 1 –12.

[42] K. Yedavalli, B. Krishnamachari, S. Ravula, and B. Srinivasan, "Ecolocation: A sequence based technique for RF localization in wireless sensor networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, 2005, pp. 285–292.

[43] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell, "SoundSense: Scalable sound sensing for people-centric applications on mobile phones," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, 2009, pp. 165–178.

[44] A. Varshavsky, E. de Lara, J. Hightower, A. LaMarca, and V. Otsason, "GSM indoor localization," *Pervasive and Mobile Computing, Elsevier*, vol. 3, no. 6, pp. 698–720, 2007.

[45] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha, "FM-based indoor localization," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys '12)*, 2012, pp. 169–182.

[46] A. Popleteev, V. Osmani, and O. Mayora, "Investigation of indoor localization with ambient FM radio stations," in *Proceedings of the 10th IEEE International Conference on Pervasive Computing and Communications (PerCom '12)*, 2012, pp. 171–179.

[47] N. Patwari and J. Wilson, "RF sensor networks for device-free localization: Measurements, models, and algorithms," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1961–1973, 2010.

[48] M. Youssef, M. Mah, and A. Agrawala, "Challenges: Device-free passive localization for wireless environments," in *Proceedings of the 13th ACM International Conference on Mobile Computing and Networking (MobiCom '07)*, 2007, pp. 222–229.

[49] M. Moussa and M. Youssef, "Smart devices for smart environments: Device-free passive detection in real environments," in *Proceedings of the 7th IEEE International Conference on Pervasive Computing and Communications (PerCom '09)*, 2009, pp. 1–6.

[50] J. Wilson and N. Patwari, "See-through walls: Motion tracking using variance-based radio tomography networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 5, pp. 612–621, 2011.

[51] C. Chang and A. Sahai, "Object tracking in a 2D UWB sensor network," in *Conference Record of the 38th Asilomar Conference on Signals, Systems and Computers*, vol. 1, 2004, pp. 1252–1256.

[52] E. Paolini, A. Giorgetti, M. Chiani, R. Minutolo, and M. Montanari, "Localization capability of cooperative anti-intruder radar systems," *EURASIP Journal on Advances in Signal Processing*, pp. 1–14, 2008.

[53] Q. Hao, D. Brady, B. D. Guenther, J. Burchett, M. Shankar, and S. Feller, "Human tracking with wireless distributed pyroelectric sensors," *IEEE Sensors Journal*, vol. 6, no. 6, pp. 1683–1696, 2006.

[54] B. Song, H. Choi, and H. S. Lee, "Surveillance tracking system using passive infrared motion sensors in wireless sensor network," in *Proceedings of the 2008 International Conference on Information Networking (ICOIN '08)*, 2008, pp. 1–5.

[55] P. Zappi, E. Farella, and L. Benini, "Tracking motion direction and distance with pyroelectric IR sensors," *IEEE Sensors Journal*, vol. 10, no. 9, pp. 1486–1494, 2010.

[56] A. Fod, A. Howard, and M. Mataric, "A laser-based people tracker," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA '02)*, vol. 3, 2002, pp. 3024–3029.

[57] H. Zhao and R. Shibasaki, "A novel system for tracking pedestrians using multiple single-row laser-range scanners," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 2, pp. 283–291, 2005.

[58] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE Transactions Pattern Analysis Machine Intelligence*, vol. 31, no. 12, pp. 2179–2195, 2009.

[59] T. Zhao and R. Nevatia, "Tracking multiple humans in crowded environment," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, vol. 2, 2004, pp. 406–413.

[60] K. Okuma, A. Taleghani, N. Freitas, J. Little, and D. Lowe, "A boosted particle filter: Multitarget detection and tracking," in *Proceedings of the 8th European Conference on Computer Vision (ECCV '04)*, 2004, pp. 28–39.

[61] J. Giebel, D. Gavrila, and C. Schnrr, "A bayesian framework for multi-cue 3D object tracking," in *Proceedings of the 8th European Conference on Computer Vision (ECCV '04)*, 2004, pp. 241–252.

[62] K. Smith, D. Gatica-Perez, and J.-M. Odobez, "Using particles to track varying numbers of interacting people," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005, pp. 962–969.

[63] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua, "Multicamera people tracking with a probabilistic occupancy map," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 267–282, 2008.

[64] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz, "Localization from connectivity in sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 11, pp. 961–974, 2004.

[65] R. Nagpal, H. Shrobe, and J. Bachrach, "Organizing a global coordinate system for local information on an ad hoc sensor network," in *Proceedings of the 2nd International Conference on Information Processing in Sensor Networks (IPSN '03)*, 2003, pp. 333–348.

[66] M. Li and Y. Liu, "Rendered path: Range-free localization in anisotropic sensor networks with holes," in *Proceedings of the 13th International Conference on Mobile Computing and Networking (MobiCom '07)*, 2008, pp. 51–62.

[67] H. T. Kung, C.-K. Lin, T. han Lin, and D. Vlah, "Localization with snap-inducing shaped residuals (SISR): Coping with errors," in *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking (MobiCom '09)*, 2009, pp. 333–344.

[68] R. Huang and G. V. Záruba, "Incorporating multiple sensory data for mobile ad hoc networks localization," *IEEE Transactions on Mobile Computing*, vol. 6, no. 9, pp. 1090–1104, 2007.

[69] L.-w. Chan, J.-r. Chiang, Y.-c. Chen, C.-n. Ke, J. Hsu, and H.-h. Chu, "Collaborative localization: Enhancing WiFi-based position estimation with neighborhood links in clusters," in *Proceedings of the 4th International Conference on Pervasive Computing (Pervasive '06)*, 2006, pp. 50–66.

[70] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan, "BeepBeep: A high accuracy acoustic ranging system using COTS mobile devices," in *Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems (SenSys '07)*, 2007, pp. 1–14.

[71] J. Qiu, D. Chu, X. Meng, and T. Moscibroda, "On the feasibility of real-time phone-to-phone 3D localization," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems(SenSys '11)*, 2011, pp. 190–203.

[72] B. Krach and P. Robertson, "Integration of foot-mounted inertial sensors into a bayesian location estimation framework," in *Proceedings of the 5th International Workshop on Positioning, Navigation and Communication (WPNC '08)*, 2008, pp. 55–61.

[73] U. Steinhoff and B. Schiele, "Dead reckoning from the pocket — an experimental study," in *Proceedings of the 8th International Conference on Pervasive Computing and Communications (PerCom '10)*, 2010, pp. 162–170.

[74] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *Proceedings of the 10th International Conference on Ubiquitous Computing (UbiComp '08)*, 2008, pp. 114–123.

[75] H. Wang, H. Lenz, A. Szabo, J. Bamberger, and U. Hanebeck, "WLAN-based pedestrian tracking using particle filters and low-cost MEMS sensors," in *Proceedings of the 4th Workshop on Positioning, Navigation and Communication (WPNC '07)*, 2007, pp. 1–7.

[76] Y. Jin, W.-S. Soh, M. Motani, and W.-C. Wong, "A robust indoor pedestrian tracking system with sparse infrastructure support," *IEEE Transactions on Mobile Computing*, vol. 12, no. 7, pp. 1392–1403, 2013.

[77] H. T. Kung and D. Vlah, "Efficient location tracking using sensor networks," in *Proceedings of the 2003 IEEE Wireless Communications and Networking Conference (WCNC '03)*, vol. 3, 2003, pp. 1954–1961.

[78] W. Zhang and G. Cao, "DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks," *IEEE Transactions on Wireless Communications*, vol. 3, no. 5, pp. 1689–1701, 2004.

[79] ——, "Optimizing tree reconfiguration for mobile target tracking in sensor networks," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 4, 2004, pp. 2434–2445.

[80] W.-P. Chen, J. C. Hou, and L. Sha, "Dynamic clustering for acoustic target tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 3, pp. 258–271, 2004.

[81] X. Ji, H. Zha, J. Metzner, and G. Kesidis, "Dynamic cluster structure for object detection and tracking in wireless ad-hoc sensor networks," in *Proceedings of the 2004 IEEE International Conference on Communications (ICC '04)*, vol. 7, 2004, pp. 3807–3811.

[82] Y. Xu, J. Winter, and W.-C. Lee, "Prediction-based strategies for energy saving in object tracking sensor networks," in *Proceedings of the 2004 IEEE International Conference on Mobile Data Management (MDM '04)*, 2004, pp. 346–357.

[83] ——, "Dual prediction-based reporting for object tracking sensor networks," in *Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems, Networking and Services (Mobiquitous '04)*, 2004, pp. 154–163.

[84] M. B. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær, "EnTracked: Energy-efficient robust position tracking for mobile devices," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, 2009, pp. 221–234.

[85] I. Constandache, S. Gaonkar, M. Sayler, R. Choudhury, and L. Cox, "Enloc: Energy-efficient localization for mobile phones," in *Proceedings of the 28th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '09)*, 2009, pp. 2716–2720.

[86] J. Paek, J. Kim, and R. Govindan, "Energy-efficient rate-adaptive GPS-based positioning for smartphones," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, 2010, pp. 299–314.

[87] Scalable Network Technologies, "Qualnet," [Online]. Available: http://www.scalable-networks.com/products/qualnet/.

[88] A. Savvides, W. L. Garber, R. L. Moses, and M. B. Srivanstava, "An analysis of error inducing parameters in multihop sensor node localization," *IEEE Transactions on Mobile Computing*, vol. 4, no. 2, pp. 567–577, 2005.

[89] M. Wallbaum and S. Diepolder, "A motion detection scheme for wireless LAN stations," in *Proceedings of 3rd International Conference on Mobile Computing and Ubiquitous Networking (ICMU '06)*, 2006, pp. 2–9.

[90] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *SIGKDD Explor. Newsl.*, vol. 12, no. 2, pp. 74–82, 2010.

[91] T. Choudhury, S. Consolvo, B. Harrison, J. Hightower, A. LaMarca, L. LeGrand, A. Rahimi, A. Rea, G. Bordello, B. Hemingway, P. Klasnja, K. Koscher, J. Landay, J. Lester, D. Wyatt, and D. Haehnel, "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Computing*, vol. 7, no. 2, pp. 32–41, 2008.

[92] P. Lukowicz, S. Pentland, and A. Ferscha, "From context awareness to socially aware computing," *IEEE Pervasive Computing*, vol. 11, no. 1, pp. 32–41, 2012.

[93] S. Chitte, S. Dasgupta, and Z. Ding, "Distance estimation from received signal strength under log-normal shadowing: Bias and variance," *IEEE Signal Processing Letters*, vol. 16, no. 3, pp. 216–218, 2009.

[94] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proceedings of the 24th International Conference on Management of Data (SIGMOD '05)*, 2005.

[95] E. M. Knorr, R. T. Ng, and V. Tucakov, "Distance-based outliers: Algorithms and applications," *The VLDB Journal*, vol. 8, no. 3-4, pp. 237–253, 2000.

[96] J.-g. Park, A. Patel, D. Curtis, S. Teller, and J. Ledlie, "Online pose classification and walking speed estimation using handheld devices," in *Proceedings of the 14th ACM International Conference on Ubiquitous Computing (UbiComp '12)*, 2012, pp. 113–122.

[97] T. Higuchi, H. Yamaguchi, and T. Higashino, "Clearing a crowd: Cotext-supported neighbor positioning for people-centric navigation," in *Proceedings of the 10th International Conference on Pervasive Computing (Pervasive '12)*, 2012, pp. 325–342.

[98] X. Hong, M. Gerla, G. Pei, and C.-C. Chiang, "A group mobility model for ad hoc wireless networks," in *Proceedings of the 2nd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '99)*, 1999, pp. 53–60.

[99] B. Peterson, R. Baldwin, and J. Kharoufeh, "Bluetooth inquiry time characterization and selection," *IEEE Transactions on Mobile Computing*, vol. 5, no. 9, pp. 1173–1187, 2006.

[100] J. C. S. Jacques Junior, S. Musse, and C. Jung, "Crowd analysis using computer vision techniques," *IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 66–77, 2010.

[101] J. Weppner and P. Lukowicz, "Collaborative crowd density estimation with mobile phones," in *Proceedings of the 2nd International Workshop on Sensing Applications on Mobile Phones (PhoneSense '11)*, 2011.

[102] ——, "Bluetooth based collaborative crowd density estimation with mobile phones," in *Proceedings of the 11th IEEE International Conference on Pervasive Computing and Communications (PerCom '13)*, 2013, pp. 193–200.