URBAN DIGITAL MAP CONSTRUCTION AND UPDATE FOR LOCATION BASED SERVICE IN MOBILE SENSING

ZHE PENG

PhD

The Hong Kong Polytechnic University

2018

THE HONG KONG POLYTECHNIC UNIVERSITY
DEPARTMENT OF COMPUTING

URBAN DIGITAL MAP CONSTRUCTION AND UPDATE FOR LOCATION BASED SERVICE IN MOBILE SENSING

Zhe PENG

A thesis submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
May 2018

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

Signature of Author Zhe PENG

Abstract

The inherent limitations in accuracy and diversity of urban digital maps are great obstacles to providing various location based services to mobile users. The emergence of mobile sensing, combining the power of both mobile computing and intelligent sensing, heralds a promising solution to the limitations on urban digital maps. In mobile sensing, urban digital maps are constructed and updated from multiple sensing data collected from mobile devices. The main disadvantage associated with this new technology is that it is difficult to extract accuracy and valuable information from a large amount of sensing data to construct and update urban digital maps. Therefore, it is crucial to design automatic indoor floor plan construction method, outdoor map update method, and urban safety index map construction method to support various location based services.

In this thesis, we first propose the PlanSketcher system architecture to construct fine-grained and facility-labelled indoor floor plans with less energy consumption in smartphone. New landmark recognition approach is proposed to detect various landmarks. Then hallway topologies are constructed based on the sensing data, depth data and images through the proposed traverse-independent hallway construction algorithms. Because the indoor facility information is a crucial component of the indoor floor plan, we construct the room shape and label the recognized facilities in their

corresponding positions to generate a complete indoor floor plan.

To update the outdoor map, we propose an automatic store self-updating system through street views and sensing data crowdsourced from mobile users. A new weighted artificial neural network is developed to learn the underlying relationship between estimated positions and real positions to localize user's shooting positions. Then, we design a novel store name recognition method by considering two valuable features (i.e. the position of text in image and the colour histogram). In this way, we are able to recognize the complete store name instead of individual letters as previous study. Furthermore, we transfer the shooting position to the location of recognized stores in the map. To update changed stores in the map, we consider three updating categories (replacing, adding, and deleting) and estimate their positions based on the kernel density estimate model.

Finally, to support urban safety related services, we propose an urban safety analysis system to infer safety index by leveraging multiple cross-domain urban location-based data. Effective spatially-related and temporally-related features are extracted from various data, including urban map, housing rent and density, population, positions of police stations, point of interests (POIs), crime event records, and taxi GPS trajectories. Then we present a novel feature fusion method to feed fused features into a spatial or temporal classifier, instead of treating features equally, leading to a high classification and inference accuracy. In addition, we design a new co-training-based learning method to accurately infer the safety index of each position in a city. The approach consists of two classifiers respectively modelling the spatial and temporal features which both influence the safety index.

Publications

Journal Articles

1. **Zhe Peng**, Shang Gao, Bin Xiao, Songtao Guo, and Yuanyuan Yang, CrowdGIS: Updating Digital Maps via Mobile Crowdsensing, accepted in *IEEE Transactions on Automation Science and Engineering (T-ASE)*, Vol. 15, No. 1, Jan. 2018, pp. 369-380.

2. **Zhe Peng**, Shang Gao, Zecheng Li, Bin Xiao, and Yi Qian, Vehicle Safety Improvement through Deep Learning and Mobile Sensing, accepted in *IEEE Network*, Vol. 32, No. 4, July 2018, pp. 28-33.

3. **Zhe Peng**, Shang Gao, Bin Xiao, Guiyi Wei, Songtao Guo, and Yuanyuan Yang, Indoor Floor Plan Construction through Sensing Data Collected from Smartphones, accepted in *IEEE Internet of Things Journal (IoT-J)*, 2018.

4. Jiwei Li, **Zhe Peng**, Shang Gao, Bin Xiao, and Henry Chan, Smartphone-Assisted Energy Efficient Data Communication for Wearable Devices, accepted in *Computer Communications (Elsevier)*, Vol. 105, June 2017, pp. 33-43.

Conference Papers

1. **Zhe Peng**, Bin Xiao, Yuan Yao, Jichang Guan, and Fan Yang, U-Safety: Urban Safety Analysis in A Smart City, in *Proc. of the IEEE International Conference on Communications (ICC)*, Paris, France, 21-25 May 2017, pp. 1-6.

2. Jiwei Li, **Zhe Peng**, and Bin Xiao, New Mobility-aware Application Offloading Design with Low Delay and Energy Efficiency, in *Proc. of the IEEE International Conference on Communications (ICC)*, Kansas City MO, USA, 20-24 May 2018, pp. 1-6.

3. Shang Gao, **Zhe Peng**, Bin Xiao, Aiqun Hu, and Kui Ren, FloodDefender: Protecting Data and Control Plane Resources under SDN-aimed DoS Attacks, in *Proc. of the IEEE International Conference on Computer Communications (INFOCOM)*, Atlanta GA, USA, 1-4 May 2017, pp. 1-9.

4. Shang Gao, **Zhe Peng**, Bin Xiao, Qingjun Xiao, and Yubo Song, SCoP: Smartphone Energy Saving by Merging Push Services in Fog Computing, in *Proc. of IEEE/ACM International Symposium on Quality of Service (IWQoS)*, Vilanova i la Geltru, Spain, 14-16 June 2017, pp. 1-10.

5. Yuan Yao, **Zhe Peng**, Bin Xiao, and Jichang Guan, An Efficient Learning-based Approach to Multi-objective Route Planning in a Smart City, in *Proc. of the IEEE International Conference on Communications (ICC)*, Paris, France, 21-25 May 2017, pp. 1-6.

6. Shang Gao, **Zhe Peng**, Bin Xiao, and Yubo Song, Secure and Energy Efficient

Prefetching Design for Smartphones, in *Proc. of the IEEE International Conference on Communications (ICC)*, Kuala Lumpur, Malaysia, 23-27 May 2016, pp. 1-6.

7. Jiwei Li, **Zhe Peng**, and Bin Xiao, Smartphone-Assisted Smooth Live Video Broadcast on Wearable Cameras, in *Proc. of IEEE/ACM International Symposium on Quality of Service (IWQoS)*, Beijing, China, 20-21 June 2016, pp. 1-6.

8. Jiwei Li, **Zhe Peng**, Bin Xiao, and Yu Hua, Make Smartphones Last A Day: Pre-processing Based Computer Vision Application Offloading, in *Proc. of the IEEE Communications Society Conference on Sensing, Communication and Networking (SECON)*, Seattle WA, USA, 22-25 June 2015, pp. 1-9.

Acknowledgements

First and foremost, I would like to thank my supervisor, Dr. Bin Xiao, for giving me inspirations on research ideas and tremendous support on conducting research. His advice on problem formulation and methodology has given me abundant confidence in tackling difficult research problems, and his suggestions of formal English writing are proved to be significantly useful. His advice and suggestions have helped me become a better researcher, and will continue to do so.

I would also like to thank Prof. Wei Lou, Prof. Yuanqing Zheng, and our previous and current group members (Dr. Qingjun Xiao, Dr. Kai Bu, Dr. Xuan Liu, Dr. Jiwei Li, Shang Gao, and Zecheng Li) for contributing, directly or indirectly, to some chapters of the thesis. Every conversation with them motivates and inspires me to conduct high quality research and their selfless help is of great value to my research. I am also grateful to friends for their company and encouragement. I especially would like to thank Shuhang Gu, Wenjian Xu, Feng Tan, and Yanxing Hu for their emotional support and selfless help.

Finally, and most importantly, I would like to thank my wife, parents and relatives for their unconditional love and support. The thesis is dedicated to them all.

Hong Kong S.A.R., China Zhe PENG

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Traditional Urban Digital Map Construction and Update Techniques for Location Based Service

During the last decade, various mobile devices, such as smartphones and wearable devices, have become widely popular and changed the way people live and move. Driven by the flourishing of mobile devices, there has been a rapid growth in location based services, including localization, navigation, route planning, finding point-of-interests, geo-social network, and advertisement. Location based services (LBS) can be defined as services that depend on and are enhanced by positional information of mobile device [Dhar and Varshney, 2011]. A location based service is a mobile information service that extends spatial and temporal information processing capability to end users via Internet and wireless communications. Figure 1.1 gives an illustration of location based service. At the absence of personal computers, people rely on mobile devices as the main means to obtain these location based services. Nowadays, obtaining location based services has become a fundamental need for many people.

In order to provide satisfactory location based services to users, a corresponding

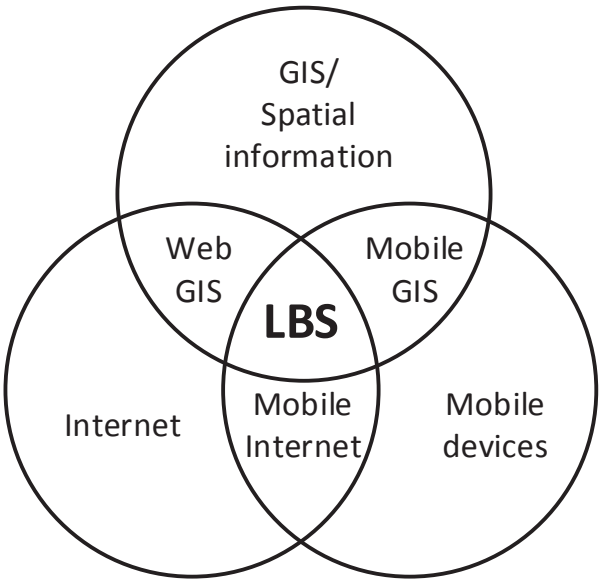


Fig. 1.1: The illustration of location based service

urban digital map is the critical foundation. This map can be a floor plan in the case of indoor environment, or a street map in the case of outdoor environment. These urban digital maps can provide abundant location related information, including the structure of roads, the positions of point-of-interests, and extra details about point-of-interests. Traditionally, urban digital maps can be constructed and updated by collecting indoor floor plans from building owners or hiring dedicated personnel to gather data.

Although these methods can construct and update urban digital maps, neither is conducive to large scale coverage in short time. For instance, service providers have to conduct effort-intensive and time-consuming business negotiations with building operators to collect the floor plans, or wait for them to voluntarily upload such data. And many legacy buildings may not have floor plans at all. Constructing and updating maps by hiring dedicated personnel also needs extensive manual calibrations.

Moreover, these methods cannot construct some special maps, such as urban safety index map which provides the safety index of each area in the city.

1.2 The Emergence of Mobile Sensing

With the continuous improvement of computational power in mobile devices, more and more built-in sensors have been equipped, which include accelerometer, gyroscope, digital compass, GPS, microphone, camera, etc. Recent development in mobile computing technologies greatly accelerates the realization of the ideal of mobile sensing. Mobile sensing denotes the use of mobile devices and their embedded sensors for sensing and learning physical and social phenomena, and using derived information to inform, share, and persuade humans [Khan et al., 2013, Lane et al., 2010]. with the mobile sensing technique, many new applications have emerged across a wide variety of domains, such as localization, healthcare, social networks, safety, environmental monitoring, and transportation.

For the location based services, mobile sensing technique can be leveraged to construct and update various urban digital maps. Individual mobile devices collect different sensing data from embedded sensors and transfer them to the server through wireless transmission. These sensing data include acceleration, rotational velocity, magnetic field, GPS, WiFi data, Bluetooth data, images, etc. Then various maps information, such as hallways, rooms, roads, and buildings, are extracted from the sensing data by applying many data analysis techniques. Finally, extracted maps information are fed to the corresponding location based services or informed to the public. Thus, the mobile sensing technique opens the door for effortless and effective construction and update of digital maps, such as indoor map, outdoor map, and

urban safety index map.

1.3 Problem Statement

Although the mobile sensing technique has shown significant advantages over traditional methods to construct and update urban digital maps, there is still some space for the improvement. One problem associated with this new technology is that it needs a large amount of mobile users to traverse all hallways and corners to collect sensing data when constructing the indoor maps. And the room names cannot be recognized correctly and added in the maps, which are extremely crucial for practical location based services. For outdoor environment, stores may be changed in the city, and it is difficult to timely update these changed stores in the outdoor maps. Moreover, the safety information in the city is important for travellers to plan their routes. Because the safety index is influenced by multiple factors with a nonlinear relationship, it is not trivial to predict the safety index of each position and construct the safety index map. These problems have motivated us to develop more effective urban digital map construction and update approaches to support various location based services.

In this thesis, we focus on three topics associated with the urban digital map: indoor floor plan construction, outdoor map update, and urban safety index map construction.

- Indoor floor plan construction. The goal of indoor floor plan construction is to provide the foundation of flourishing indoor location-based services for smartphone, such as indoor localization and navigation. This topic is very challenging because a general solution for the optimal generation of a fine-grained indoor

floor plan based on the natural state of the environment is unavailable yet. Traditional methods of indoor floor plan construction consume intensive labour and time, and cannot provide precise information about facilities. In this thesis, we first focus on how to enable one user to construct fine-grained and facility-labelled indoor floor plans.

- Outdoor map update. The second topic we focus on is an automatic store updating system for outdoor digital maps that leverages street views and sensing data crowdsourced from mobile users. Existing map update approaches can only update stores through a manual survey or update roads automatically from mobile crowdsensing data. Since the store information is a crucial component in the digital map, a novel automatic store updating method should be proposed to timely update stores in the map.

- Urban safety index map construction. Urban safety information of an area, such as the occurrence of crime and emergency, is of great importance to support urban safety control and protect humans from danger. A traveller urgently needs such valuable information to book a hotel before his arrival, and to choose a safe walking route in a time slot when he stays in a new and unfamiliar city. Existing methods can only roughly assess and predict the safety index of an area in a city. However, these methods cannot unveil the impact of multiple factors and thus cannot present accurate safety level of an area nor predict it. Therefore, the third topic we focus on is an urban safety analysis system to infer safety index by leveraging multiple cross-domain urban location-based data.

1.4 Thesis Contributions

In this thesis, we design feasible approaches that construct and update urban digital maps to provide better location based services to mobile users. We attempt to solve the three identified challenges aforementioned, as detailed as follows.

1.4.1 Indoor Floor Plan Construction through Mobile Sensing

With the development of sensing technology, smartphones can provide various kinds of data, including inertial sensing data, WiFi data, depth data and images. These data make it possible to construct accurate indoor floor plans that are the critical foundations of flourishing indoor location-based services for smartphone. However, even with the popular crowdsourcing approach, the wide construction of indoor floor plans has not yet to be realized due to the intensive time consumption.

In Chapter 3, we utilize machine learning techniques to build PlanSketcher, a system that enables one user to construct fine-grained and facility-labelled indoor floor plans accurately. First, the proposed system extracts novel integrated features to recognize diverse landmarks. Second, traverse-independent hallway topologies are constructed based on the sensing data, depth data and images through the proposed hallway construction algorithms. Finally, PlanSketcher constructs the room shape and labels recognized facilities in their corresponding positions to generate a complete indoor floor plan. Because PlanSketcher exploits different kinds of data collected from smartphones with new feature extraction method, it can obtain accurate indoor floor plan topology and facility labels. We implement PlanSketcher and conduct extensive experiments in 3 large indoor settings. The evaluation results illustrate that

PlanSketcher outperforms the state-of-the-art methods by showing smaller position and orientation error, more recognized facilities and less energy consumption.

1.4.2 Updating Digital Maps via Mobile Crowdsensing

Accurate digital maps with abundant store information play a crucial role in various location-based services and applications. Based on real investigations, current digital maps still lack a large amount of store information. Replaced stores and nonexistent stores cannot be timely updated in the map neither. Existing map update approaches can only update stores through a manual survey or update roads automatically from mobile crowdsensing data. Thus, the problem of updating stores in digital map has not been studied well and remains open.

In Chapter 4, we propose CrowdGIS, an automatic store self-updating system for digital maps that leverages street views and sensing data crowdsourced from mobile users. We first develop a new weighted artificial neural network to learn the underlying relationship between estimated positions and real positions to localize user's shooting positions. Then, a novel text detection method is designed by considering two valuable features, including the color and texture information of letters. In this way, we can recognize complete store name instead of individual letters as in the previous study. Furthermore, we transfer the shooting position to the location of recognized stores in the map. Finally, CrowdGIS considers three updating categories (replacing, adding, and deleting) to update changed stores in the map based on the kernel density estimate model. We implement CrowdGIS and conduct extensive experiments in a real outdoor region for 1 month. The evaluation results demonstrate that CrowdGIS effectively accommodates store variations and updates stores to maintain an up-to-date map with high accuracy.

1.4.3 Urban Safety Index Inference from Location-based Data

Information about urban safety, e.g., the safety index of a position, is of great importance to protect humans and support safe walking route planning. Despite some research on urban safety analysis, the accuracy and granularity of safety index inference are both very limited. The problem of analyzing urban safety to predict safety index throughout a city has not been sufficiently studied and remains open.

In Chapter 5, we propose U-Safety, an urban safety analysis system to infer safety index by leveraging multiple cross-domain urban location-based data. We first extract spatially-related and temporally-related features from various urban location-based data, including urban map, housing rent and density, population, positions of police stations, point of interests (POIs), crime event records, and taxi GPS trajectories. Then, these features are fed into a novel sparse auto-encoder (SAE) framework with feature correlation constraint to obtain the final discriminative feature representation. Finally, we design a new co-training-based learning method, which consists of two separated classifiers, to calculate safety index accurately. We implement U-Safety and conduct extensive experiments by utilizing various real data sources obtained in New York City. The evaluation results demonstrate the advantages of U-Safety over other methods.

1.5 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 introduces the related work. Chapter 3 utilizes machine learning techniques to build PlanSketcher, a system that enables one user to construct fine-grained and facility-labelled indoor floor plans accurately. The proposed system outperforms the state-of-the-art methods by

showing smaller position and orientation error, more recognized facilities and less energy consumption. Chapter 4 proposes an automatic store self-updating system for digital maps that leverages street views and sensing data crowdsourced from mobile users. The proposed approach can effectively accommodate store variations and update stores to maintain an up-to-date map with high accuracy. Chapter 5 proposes an urban safety analysis system to infer safety index by leveraging multiple cross-domain urban location-based data. The proposed approach can dynamically predict the safety index of each area in smart city with high fine granularity. Finally, Chapter 6 concludes the thesis and indicates future work.

The primary research outputs emerged from the thesis are as follows:

- Zhe Peng, Shang Gao, Bin Xiao, Songtao Guo, and Yuanyuan Yang, CrowdGIS: Updating Digital Maps via Mobile Crowdsensing, accepted in *IEEE Transactions on Automation Science and Engineering (T-ASE)*, Vol. 15, No. 1, Jan. 2018, pp. 369-380.

- Zhe Peng, Shang Gao, Zecheng Li, Bin Xiao, and Yi Qian, Vehicle Safety Improvement Through Deep Learning and Mobile Sensing, accepted in *IEEE Network*, Vol. 32, No. 4, July 2018, pp. 28-33.

- Zhe Peng, Shang Gao, Bin Xiao, Guiyi Wei, Songtao Guo, and Yuanyuan Yang, Indoor Floor Plan Construction through Sensing Data Collected from Smartphones, accepted in *IEEE Internet of Things Journal (IoT-J)*, 2018.

- Zhe Peng, Bin Xiao, Yuan Yao, Jichang Guan, and Fan Yang, U-Safety: Urban Safety Analysis in A Smart City, in *Proc. of the IEEE International Conference on Communications (ICC)*, Paris, France, 21-25 May 2017, pp. 1-6.

- Zhe Peng, Shang Gao, Bin Xiao, Songtao Guo, and Yuanyuan Yang, When Urban Safety Index Inference Meets Location-based Data, under review in *IEEE Transactions on Mobile Computing (TMC)*.

Chapter 2

Literature Review

The unprecedented development in both mobile computing and intelligent sensing has spiked strong interests in developing systems that combine both [Lane et al., 2010, Yürür et al., 2016]. This has resulted in the rise of mobile sensing, which shows great potential in resolving the inherent constraints in location based services. From a high-level perspective, the study of mobile sensing can be categorized into three major topics, i.e., indoor floor plan construction, outdoor map update, and urban safety index map construction. This chapter presents a comprehensive literature review of these topics in mobile sensing.

2.1 Indoor Floor Plan Construction

2.1.1 Inertia-based Sensing

Indoor floor plan construction is an extensively studied field in mobile sensing. Most of the existing approaches focus primarily on inertial sensing data aggregations. CrowdInside [Alzantot and Youssef, 2012] is a crowdsourcing-based system for automatically constructing indoor floor plans. It leverages various inertial sensing data collected from the smartphones to generate user motion traces. Some landmarks such

as elevators, stairs and locations with GPS reception are recognized to calibrate users' trajectories and accumulative errors. The proposed algorithm is highly dependent on the accuracy of crowdsourced motion traces. Jiang *et al.* [Jiang et al., 2013] leverage WiFi fingerprints and user motion information to obtain the room sizes and hallway orientations to automatically construct indoor floor plans. Walkie-Markie [Shen et al., 2013] uses the locations at which the trend of WiFi signal strength changes as the landmarks to construct indoor floor plans. Such landmarks are also used to localize the user in various localization systems [Liu et al., 2012a, Shangguan et al., 2014, Zhang et al., 2014].

2.1.2 Vision-based Sensing

Combining computer vision and mobile sensing [Gao et al., 2016] provides a new research field in indoor floor plan construction. Travi-Navi [Zheng et al., 2014] integrates the images and sensor readings to generate the navigation traces. Through tracking the navigation traces, the follower can get a vision-guided navigation. CrowdMap [Chen et al., 2015b] leverages crowdsourced sensory and video data to infer user motion traces and context of the image to produce an indoor floor plan. Indoor-Crowd2D [Chen et al., 2015a] leverages crowdsourced image and sensory data for indoor panoramic map generation. GigaSight [Simoens et al., 2013] proposes a system for continuous collection of crowdsourcing videos and performing content-based searches on mobile devices. FOCUS [Jain et al., 2013] provides an algorithm to cluster and label the crowdsourcing videos through visual reconstruction and multimodal sensing. Li-KamWa *et al.* [LiKamWa et al., 2013] analyze the energy consumption of capturing image to design a quality-energy tradeoff algorithm. Jigsaw [Gao et al., 2014] combines both image and sensing data to construct the indoor floor plan, which

achieves a better performance. However, it only uses images to infer the wall segments of the room entrance and still uses aggregated user motion trace and camera position to infer the shape of room. We cannot assume all edges and corners of the room could be covered with user traces as it may be inaccessible to users (e.g. blocked by desk).

Most of the existing works are based on the crowdsourcing method, which need to collect abundant data. Although crowdsourcing-based scheme has its own advantages and is promising for practical applications, it won't be the only way for constructing indoor floor plan. The new method we proposed avoids the well-known shortcomings caused by crowdsourcing, such as the slow-start problem, privacy issue, high overhead, device heterogeneity, etc. Therefore, the design in this thesis presents a reasonable alternative to the existing works. And different from previous efforts, the proposed method combines image recognition with human motion pattern to recognize facilities. The constructed facility-labelled floor plan could provide a more user-friendly map for location-based services.

2.2 Outdoor Digital Map Update

2.2.1 from Map Construction to Update

Smartphones with various built-in sensors [Gao et al., 2017, Li et al., 2017] have been thoroughly exploited to construct maps. Recent innovations such as Walkie-Markie [Shen et al., 2013] propose to construct indoor maps through detecting various landmarks. Such landmarks can also be used to localize users and calibrate users' trajectories in the localization systems [Liu et al., 2016, Wen et al., 2015, Yang et al., 2016]. CrowdInside [Alzantot and Youssef, 2012] is a crowdsourcing-based system for automatically constructing maps. It leverages various inertial sensing data collected

from the smartphones to generate user motion traces. As these works mainly aim at constructing maps in the initializing phase, map update is orthogonal to them in focusing on updating existing maps during serving phase to cope with environment variations over time. Thus, the map update method we proposed is compatible to the map construction by using schemes in these works.

AcMu [Wu et al., 2015a] updates WiFi Received Signal Strength (RSS) of each position in a map for wireless indoor localization. By accurately pinpointing mobile devices with a trajectory matching algorithm, the system can collect real-time RSS samples when devices are static. With these reference data, the system updates the complete radio map by learning an underlying relationship of RSS dependency between different locations. Considering environmental dynamics, LEMT [Yin et al., 2008] utilizes reference points to learn a relationship of one location and its neighbors to cope with the RSS variations. Radio map can be updated with high accuracy if the reference points are densely detected. CrowdAtlas [Wang et al., 2013] deals with the road variations in the map. It uses mobile navigation app to detect significant portions of GPS traces that do not conform to the existing map. When there is sufficient exceptional traces collected, map inference algorithms can automatically update the map. Existing works consider various dynamics in maps, while the map update method we proposed copes with a novel kind of variation which updates stores in the map.

2.2.2 Relevant Map-based Systems

Location-based service is an extensively studied field in mobile computing. Most of the existing systems depend on an accuracy map to achieve high performance. OPS [Manweiler et al., 2012] localizes a distant object in the map through various sensors

in smartphones. ParkSense [Nawaz et al., 2013] is a mobile sensing system to detect available parking spots in the map. Borealis [Zhang et al., 2011] utilizes commodity smartphones to locate WiFi access points in real time. Moreover, FOLLOWME [Shu et al., 2015] is a navigation system to navigate the users to the same destination taken by an earlier traveler with an existing map. All these technologies provide various map-based services to users, while map update methods underpin a primary support for them to maintain high performance in the long term.

2.3 Urban Safety Index Map Construction

2.3.1 Safety Data Analysis

Previous urban safety analysis methods can assess the safety index in a city and make some prediction. Prediction methods can infer relationships between visual elements and city attributes [Arietta et al., 2014], and predict potential crime rate for an area [Khosla et al., 2014]. Arietta et al. [Arietta et al., 2014] present a method for predicting relationships between visual elements and city attributes such as crime rates, theft rates and danger perception from street-level images of a city. Khosla et al. [Khosla et al., 2014] propose an approach to look beyond the immediately visible urban scene using visual elements for predicting potential crime rate for an area. However, these image-based methods cannot achieve high fine granularity or prediction accuracy. Some other methods [Christin et al., 2013, Matei et al., 2001, Traunmueller et al., 2016] develop various models to describe safety index from factors like street geometry, traffic flow and human behavior, based on a number of empirical assumptions and parameters. However, these empirical assumptions and parameters might not be applicable to all urban environments. Different from existing efforts, we

propose a new method to analyze multiple cross-domain factors to accurately infer the safety index of a position from the data mining view.

2.3.2 Various Urban Maps Construction

Many traditional systems are proposed to construct various categories of urban maps, such as outdoor maps [Wang et al., 2013], indoor maps [Jiang et al., 2013], radio maps [Yin et al., 2008], and air pollution maps [Zheng et al., 2013]. CrowdAtlas [Wang et al., 2013] deals with the road variations in the outdoor map. It uses mobile navigation app to detect portions of GPS traces that do not conform to the existing map to automatically update the map. Jiang *et al.* [Jiang et al., 2013] leverage WiFi fingerprints and user motion information to obtain the room sizes and hallway orientations to automatically construct indoor maps. LEMT [Yin et al., 2008] utilizes reference points to learn a relationship of one location and its neighbors to construct and update radio map. U-Air [Zheng et al., 2013] infers the real-time air quality information throughout a city to construct a pollution map, based on a variety of data sources collected in the city. Existing work considers various categories of dynamics of urban map. However, their methods cannot be used to construct the urban safety index map. Our proposed method realizes this objective and benefits the public by visualization.

2.3.3 Urban Computing

Nowadays sensing technologies and large-scale computing infrastructures have produced a variety of big data in urban spaces. These big and heterogeneous data imply rich knowledge about a city and can help tackle many challenges our cities face today. Motivated by the opportunity of building more intelligent cities, a series of

studies have been conducted on urban computing recently. For instance, aiming at transportation system, Wang et al. estimated the travel time of any path to improve driving experiences based on the GPS trajectories of vehicles and urban map data [Wang et al., 2014]. Considering urban electricity consumption, Momtazpour et al. presented a framework to support charging and storage infrastructure design for electric vehicles (EV) [Momtazpour et al., 2012]. In social applications, Zheng et al. provided a personalized location−activity recommendation system by mining a large number of users trajectories and location-tagged comments [Zheng et al., 2010]. Various aspects of urban data analysis have been conducted, while the urban safety analysis still lacks extensive studies.

Chapter 3

Indoor Floor Plan Construction through Mobile Sensing

With the development of sensing technology, smartphones can provide various kinds of data, including inertial sensing data, WiFi data, depth data and images. These data make it possible to construct accurate indoor floor plans that are the critical foundations of flourishing indoor location-based services for smartphone. However, even with the popular crowdsourcing approach, the wide construction of indoor floor plans has not yet to be realized due to the intensive time consumption. In this chapter, we utilize machine learning techniques to build PlanSketcher, a system that enables one user to construct fine-grained and facility-labelled indoor floor plans accurately. First, the proposed system extracts novel integrated features to recognize diverse landmarks. Second, traverse-independent hallway topologies are constructed based on the sensing data, depth data and images through the proposed hallway construction algorithms. Finally, PlanSketcher constructs the room shape and labels recognized facilities in their corresponding positions to generate a complete indoor floor plan. Because PlanSketcher exploits different kinds of data collected from smartphones with new feature extraction method, it can obtain accurate indoor floor

plan topology and facility labels. We implement PlanSketcher and conduct extensive experiments in 3 large indoor settings. The evaluation results illustrate that PlanSketcher outperforms the state-of-the-art methods by showing smaller position and orientation error, more recognized facilities and less energy consumption.

3.1 Overview

Online digital maps (e.g., Google Maps) have provided great convenience for abundant outdoor location based services (LBS) [Goswami et al., 2011, Wu et al., 2015a, Yang et al., 2012], such as finding nearby point-of-interests (POIs) and navigation. However, for indoor environments where people spend over 80% of the time [Klepeis et al., 2001], such indoor maps are extremely scarce and unavailable in most buildings. This has become a huge obstacle to pervasive indoor location based services. Service providers have to conduct effort-intensive and time-consuming business negotiations with building owners or operators to collect the floor plans, or wait for them to voluntarily upload such data. Neither is conducive to large scale coverage in short time. Therefore, accurate and scalable indoor floor plan construction at low costs is urgently needed.

Driven by the flourishing of smartphones and their capability of processing diverse data, we are motivated to construct indoor floor plans by these new techniques. Through analysing sensing data gathered from smartphones, we can capture crucial geographic features of indoor environments [Li et al., 2015]. Many efforts have been devoted to construct indoor floor plans using smartphones by crowdsourcing [Ganti et al., 2011, Jiang et al., 2013, Philipp et al., 2014]. CrowdInside [Alzantot and Youssef, 2012] first utilized inertial sensing data to automatically generate user traces

and construct indoor pathways. Walkie-Markie [Shen et al., 2013] used locations where the trend of WiFi signal strength changed as the landmarks to construct indoor floor plans. Jigsaw [Gao et al., 2014] inferred the approximate structures and shapes of hallways and rooms from the inertial data and images. CrowdMap [Chen et al., 2015b] utilized sensor-rich video data from mobile users for indoor floor plan construction. Shopprofiler [Guo et al., 2014] resorted to SSIDs collected in each shop to infer the categories and names of shops.

However, there are some limitations in current indoor floor plan construction systems. First, most of the existing systems utilize crowdsourcing to collect data, which will encounter the slow-start problem. These systems often need users to install applications in their smartphones which is not preferred in consideration of privacy and security. If not enough users participate, the collected data are insufficient to construct accurate indoor floor plans. Second, to construct accurate indoor floor plans, users need to traverse all hallways and corners in indoor environment. However, hallways are often blocked by furniture or other objects. Third, the constructed indoor floor plans with existing methods often lack valuable labels to guide users. Users may not find their destinations or know the surroundings with a floor plan without facility labels. Although Shopprofiler [Guo et al., 2014] recognizes the categories and names of shops and labels them in the floor plans, the accuracy of recognition can still be improved. The reason is that some areas are unreachable and the WiFi access service is usually provided by the shopping mall which results in the same WiFi SSID in all shops. It is extremely crucial to provide a user-friendly floor plan with facility labels for location-based services. As a consequence, we ask the following question: *Is it feasible to enable a user to easily construct his own indoor floor plans by himself*

without compromising on location accuracy and facility labels?

In this chapter, we provide an affirmative answer through the systematic design and implementation of *PlanSketcher*, which utilizes machine learning technique to construct facility-labelled and highly fine-grained indoor floor plans using smartphone. Different from current schemes, the proposed PlanSketcher constructs indoor floor plans with various data collected by a single person. This manner can save extensive manpower and energy consumption to effectively construct indoor floor plans and protect the user's privacy. Utilizing the mature object detection technique, the proposed system can also provide abundant valuable labels to distinguish various facilities. Moreover, the constructed indoor floor plans with PlanSketcher are highly fine-grained (e.g., accuracy within $1m \sim 2.5m$), thus users can locate their positions more precisely and receive much better location-based services.

Compared with previous works, PlanSketcher has the following advantages: (1) Recognizing more landmarks with high accuracy from collected data to improve the accuracy of the constructed indoor floor plans. (2) Constructing more accurate hallway topology from the landmarks, depth data and images with less energy consumption. (3) Constructing more accurate room shape with less energy consumption and labeling recognized facilities precisely in the indoor floor plans.

In this chapter, we make following contributions:

- We propose the *PlanSketcher* system architecture to construct fine-grained and facility-labelled indoor floor plans with less energy consumption in smartphone.
- We develop novel *landmark recognition* approaches to detecting various landmarks from the inertial sensing data, WiFi signals and images. Adequate landmarks are recognized such that our system is reasonably accurate.

- We present *hallway construction* algorithms to construct the topology of hallways with high fine-granularity. A new method is proposed to construct hallways that a user has traversed by fusing the depth data, inertial sensing data and images. Considering the robustness of system, we also propose the corresponding measurement calibration method. In addition, a novel Non-Traversed Hallway Construction method is designed to construct the hallways which are not traversed by a user.

- We design a *labelled room construction* method to construct the room shape from depth data and label facilities in their corresponding positions to generate a complete indoor floor plan. Different from most previous works, the proposed method does not need user to traverse in the room and can achieve a high accuracy and provide users valuable room information.

- In addition, we develop the prototype and conduct extensive evaluations across 3 large complex indoor settings on a variety of mobile devices. The results illustrate that our proposed PlanSketcher outperforms the state-of-the-art methods with smaller position and orientation error, more recognized facilities and less energy consumption.

The rest of this chapter is organized as follows. We first provide the system overview of PlanSketcher in Section 3.2. Then we detail each module of our system in Section 3.3, Section 3.4, and Section 3.5. The implementations and evaluation are illustrated in Section 3.6. Finally, we conclude this chapter in Section 3.7.

3.2 System Design

PlanSketcher system leverages inertial sensing data, depth data and images collected from smartphones to construct fine-grained and facility-labelled indoor floor plans with less energy consumption. High-performance sensors integrated in modern smartphones provide abundant information to depict user motion patterns and indoor architectural structures. With the rise of Augmented Reality (AR) technique, the time-of-flight (TOF) depth camera has been equipped within more and more smartphones. The TOF depth camera can obtain the distance between the camera and the subject for each point in the image through measuring the time-of-flight of a laser light. Images captured by cameras equipped with smartphones offer luxuriant visible and valuable description about surroundings. Figure 5.1 sketches the PlanSketcher system architecture. In the following, we describe four modules of the system briefly.

Data collection. Various inertial sensing data, WiFi data, depth data and images are gathered from smartphone while users move around in the indoor environment. Smartphone sensors such as accelerometer, gyroscope and magnetometer could measure acceleration, rotational velocity and magnetic field respectively. TOF depth camera obtains the distance between the camera and the subject for each point in the image. In addition, users leverage cameras to capture images of corners and facilities.

Landmark recognition. PlanSketcher recognizes real landmarks and virtual landmarks existing in a building. We first extract novel efficient integrated features (inertial sensing data and WiFi signals) to recognize the real landmarks, which involve elevators, escalators and stairs etc. Then two photographing manners: Corner Photographing Manner (CPM) and Facility Photographing Manner (FPM) are proposed

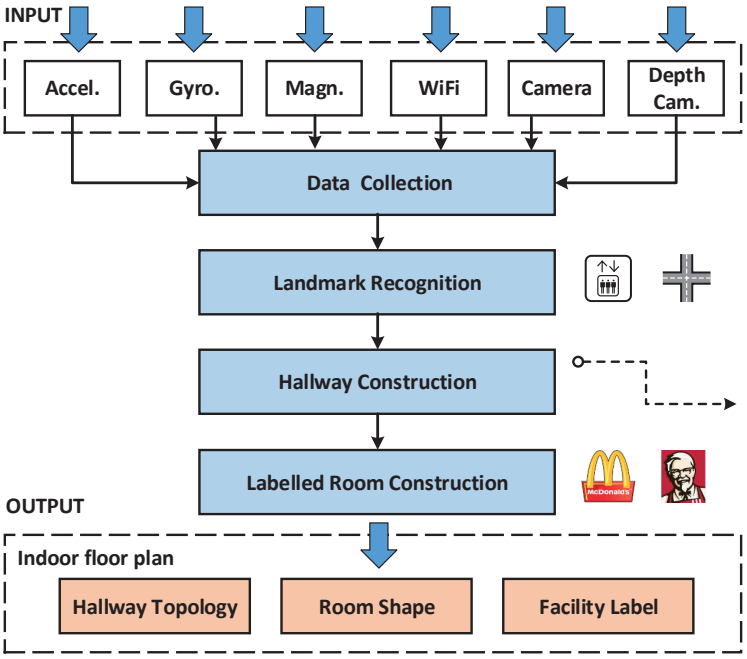


Fig. 3.1: PlanSketcher System.

to gather various information to recognize the virtual landmarks (hallway corners, stores and restrooms etc). Moreover, to distinguish various hallway corners, we define the novel and unique fingerprint for each corner through recognizing the store or room names from images with the object detection technique [Girshick et al., 2014].

Hallway construction. With the recognized landmarks, the topology of hallways is constructed based on the inertial sensing data, depth data and images. We divide the hallway construction into two types: user traversed hallways and user non-traversed hallways. A new method is proposed to construct hallways that a user has traversed by fusing the depth data, inertial sensing data and images. Considering the robustness of system, we also proposed the corresponding measurement calibration method. We propose the novel Non-traversed Hallway Construction method to construct complete floor plans through matching the fingerprints of corners, even

hallways are not traversed by the user.

Labelled room construction. To generate a relatively complete indoor floor plan, PlanSketcher constructs room shape from depth data and label recognized facilities in their corresponding positions. Different from most previous works, the user collects 3D depth data of the ceiling at the entrance instead of traversing in the room to construct room shape. Utilizing the projection of 3D depth data in the horizontal plane, PlanSketcher detects the edges and infer the geometric vertices. To label the recognized facilities (such as elevators, escalators, shops, etc), PlanSketcher calculates the position of facility entrance and its orientation.

Different from the data collection approaches adopted in prior works [Alzantot and Youssef, 2012, Guo et al., 2014, Zheng et al., 2014], our proposed system could construct accurate labelled indoor floor plans by a single person from a small amount of data. It is assumed that the participate user collects data though our defined photographing manners: Corner Photographing Manner (CPM) and Facility Photographing Manner (FPM).

3.3 Landmark Recognition

In this section, we describe the method to extract novel integrated features from human motion patterns and images to recognize various landmarks in a building. Special architectural structures, such as elevators, hallway corners and rooms, are critical to construct indoor floor plans because they describe the principal architectural features of a building. The floor plans could be constructed more accurately if these structures are recognized and located precisely.

We consider these structures as landmarks which are divided into two categories:

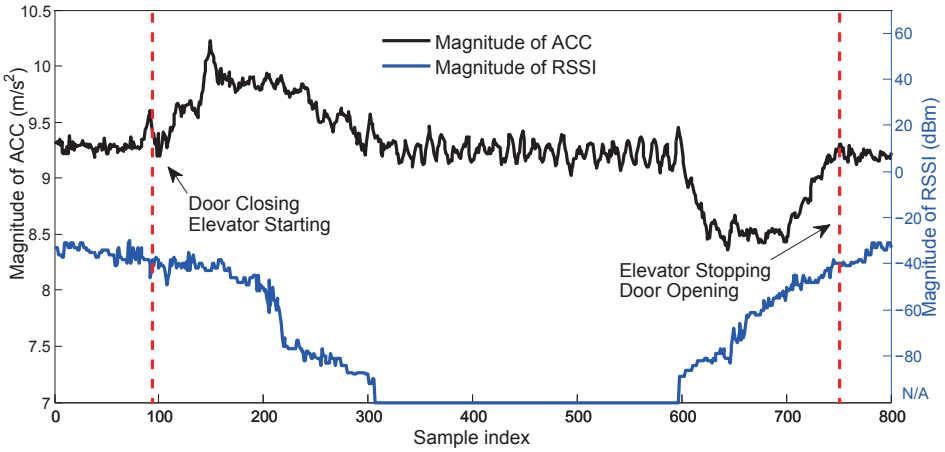


Fig. 3.2: RSSI value drops and reverses when the user enters and leaves the elevator.

real landmarks and virtual landmarks. Real landmarks compel people to move in the expectable motion patterns. Virtual landmarks are detected through matching two proposed photographing manners and recognizing the captured images.

3.3.1 Real Landmark

When people are moving within the elevators, escalators and stairs, sensing data collected by acceleration and magnetometer will present distinct features. These real landmarks are distinguished by recognizing unique features extracted from the magnitude of accelerometer and magnetometer [Alzantot and Youssef, 2012, Wang et al., 2012]. However, the recognition is erroneous (false positive or false negative) because the sensing data are easily disturbed by the unstable environment or the body's moving vibration.

To solve this problem, we combine the Received Signal Strength Indicator (RSSI) value of WiFi connection with the sensing data to recognize elevators and escalators. This idea is inspired by the observation of correlation between the RSSI value and human position. Figure 3.2 plots the differentials of RSSI value against accelerometer

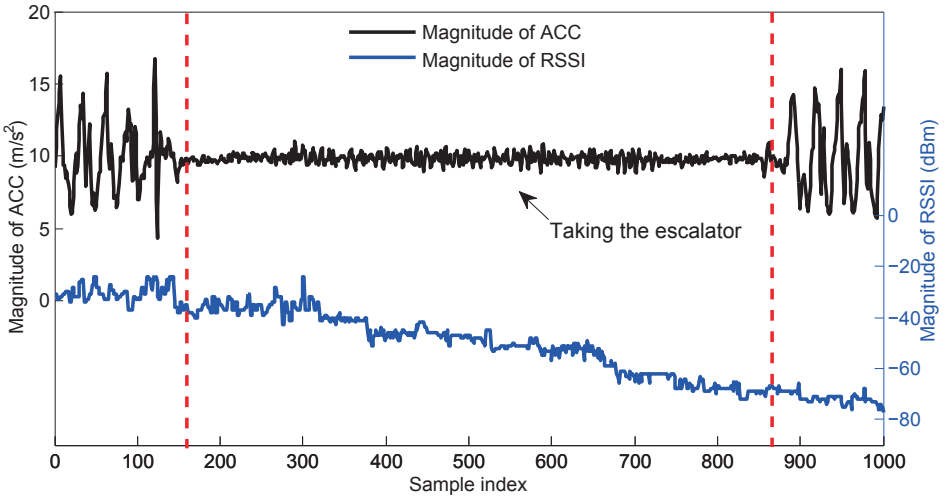
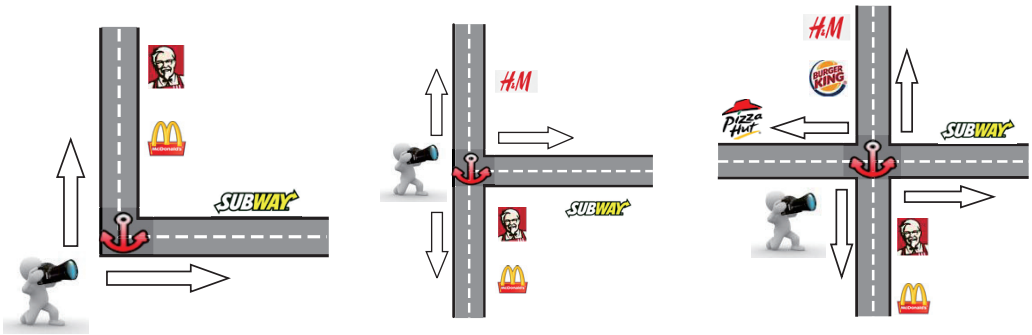


Fig. 3.3: RSSI and magnetic field changes when the user takes the escalator.

magnitude when the user takes an elevator. This figure shows that when the user enters the elevator and the door is closing, the RSSI value drops to an undetectable level due to the shielding effect of metal materials. When the elevator stops to open the door, the RSSI value reverses to the normal level. Similarly, we can also separate the escalator from the human stationary case utilizing the variance of RSSI field and magnetic field. In both cases, the accelerometer magnitude tends to be stable, but the RSSI field and magnetic field will change obviously when the user moves within the escalator (shown in Figure 3.3). Thus, by integrating the RSSI value with the sensing data, PlanSketcher is inherently more robust than existing methods which merely look into the acceleration and magnetometer reading.

3.3.2 Virtual Landmark

Hallway corners and rooms depict the basic characters of indoor environments. Because images could provide more diverse and reliable descriptions about indoor settings than the sensing data, we recognize corners and rooms from images and model



(a) A corner. (b) A T-junction. (c) A crossroad.

Fig. 3.4: Corner Photographing Manner (CPM).

them as the virtual landmarks. Two photographing manners: Corner Photographing Manner (CPM) and Facility Photographing Manner (FPM) are proposed to gather data to recognize virtual landmarks. Moreover, to distinguish various corners, we define the novel and unique fingerprint for each corner through recognizing the room names from images with the object detection technique [Girshick et al., 2014].

Corner Photographing Manner (CPM). In PlanSketcher, we design Corner Photographing Manner (CPM) which defines a few actions for the user to collect the inertial data and images of corner. As illustrated in Figure 3.4, when the user is traversing a corner, he takes a photo of the hallway with some rooms in one direction and then spins his body to photograph in another direction. During this process, the gyroscope and accelerometer offer the angular velocity and acceleration measurements, and the photographing behaviors are also recorded.

Detecting the corners from the CPM. As shown in Figure 3.5, the bumps (upward or downward) of gyroscope readings show that the user is turning left or right. During the two bumps, the user is capturing a photo of the hallway. Because the user just rotates his body during the photographing, the accelerometer magnitudes

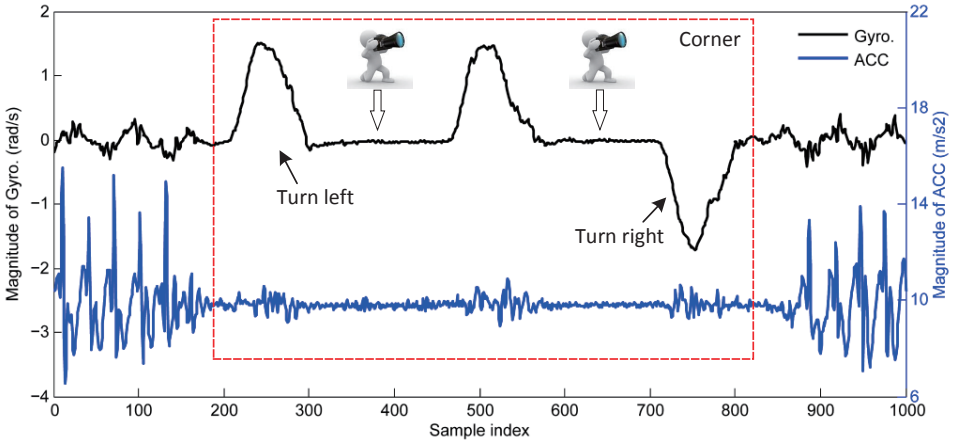


Fig. 3.5: Gyroscope and accelerometer measurements when the user photographs at the corner.

stay stable with less jitters. Triggered by the photographing behavior, the special characteristics emerged in the inertial data can indicate that the user is at the corner.

Distinguishing the corners with the fingerprints. Leveraging the output of object detection technique [Girshick et al., 2014], PlanSketcher obtains the arrays of room names from the hallway images which are viewed as the *fingerprints* of corners to distinguish various corners. Object detection is a mature technique commonly utilized to detect some specific objects from the image. Given a set of images of the target objects (e.g., the logos of various stores, alphabet and numbers) from different viewpoints, it recognizes: 1) the target rooms contained in each image; and 2) the spatial relations between these rooms. Figure 3.6 shows that two rooms are detected in the image captured at the corner. Once the room is detected, we obtain the corresponding name which is associated with the room. In addition, the closer the detected room is to the center of image, the farther it is to the photographer's location (the corner). After recognizing the rooms in each direction of the corner, we

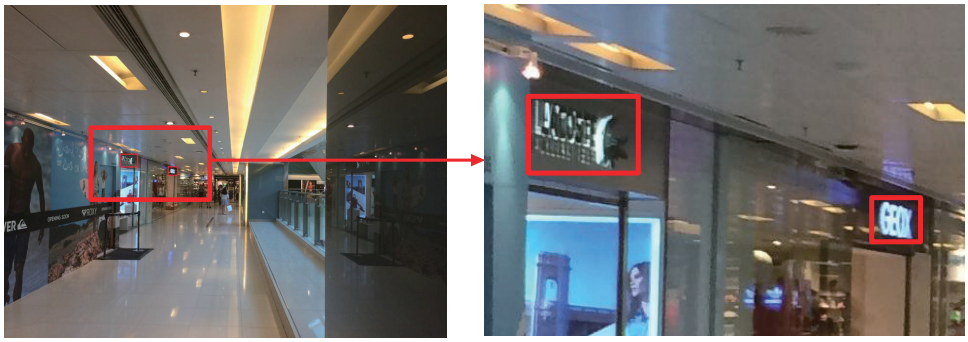


Fig. 3.6: Two stores are detected from the image captured at the corner.

define the *fingerprint* of corner C_i as

$$F_i = [f_1^i, f_2^i, \cdots, f_n^i]^T = \begin{bmatrix} N_{11}^i & N_{12}^i & \cdots & N_{1m}^i \\ N_{21}^i & N_{22}^i & \cdots & N_{2m}^i \\ \vdots & \vdots & \ddots & \vdots \\ N_{n1}^i & N_{n2}^i & \cdots & N_{nm}^i \end{bmatrix} \tag{3.1}$$

where f_n^i is sequence of room names detected in the direction n of corner C_i, and N_{nm}^i is the name of the mth room along the direction n.

Facility Photographing Manner (FPM). In PlanSketcher, we design Facility Photographing Manner (FPM) for the user to gather the inertial data and images of facilities. The facilities involve various kinds of rooms in the buildings, such as stores and restrooms etc. As illustrated in Figure 3.7, when the user is traversing a facility (such as a store), he stops and turns to face the facility to take a photo which contains the facility logo. At the same time, the gyroscope and accelerometer record the angular velocity and acceleration measurements.

Recognizing facilities from the FPM and image. Similar to the hallway corner, the facility is detected through the proposed FPM. Leveraging the object detection technique, PlanSketcher obtains the names and categories of facilities from the captured photos. During the object detection process, the training examples are

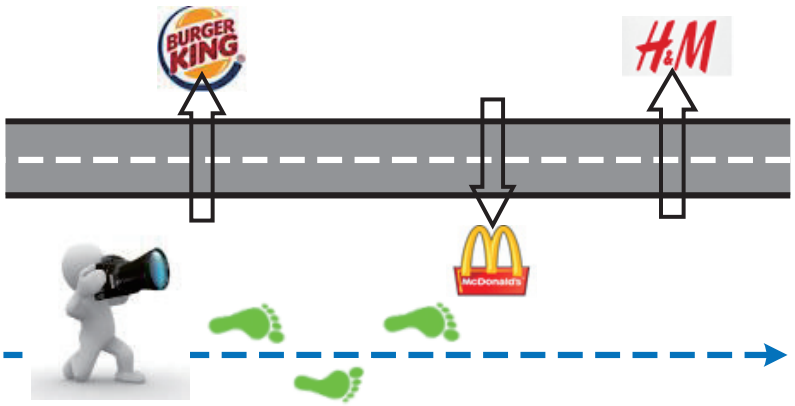


Fig. 3.7: Facility Photographing Manner (FPM).

Fig. 3.8: Facilities are recognized from the photos.

given as the triples $< name, category, images >$, which indicate the correspondences between the names and categories of target facilities and their images. The training images are gathered via photographing the facilities from different viewpoints or downloading from the Internet. Based on the observations of multiple indoor environments, we distribute the facilities into seven classes {fashion}, {supermarket}, {accessories}, {personal care}, {electronics}, {food} and {others}. Thus, when a facility is recognized from the image, the corresponding name and category are obtained. Figure 3.8 shows that facilities are recognized from the captured photos.

3.4 Hallway Construction

After recognizing various landmarks, we construct the topology of hallways based on the inertial sensing data, depth data and images. Especially, we identify two scenarios in hallway construction: 1) constructing hallways which are traversed by the user with high fine-granularity, and 2) constructing hallways which the user has not traversed. We illustrate these two scenarios as follows.

3.4.1 Constructing Traversed Hallways

It is not trivial to construct the hallway topology with high fine-granularity. Although the hallway topology could be derived from the user's trajectories tracked by the inertial sensors [Zhou et al., 2014], the accuracy of construction is relatively coarse. Thus, for hallways traversed by a user, we propose a novel method to construct the hallway utilizing the inertial sensing data, depth data and images.

To formally define this problem, we represent a hallway segment and a corner as a line segment and a cross point, and denote them as $\overline{C_iC_j}$ and C, respectively (shown

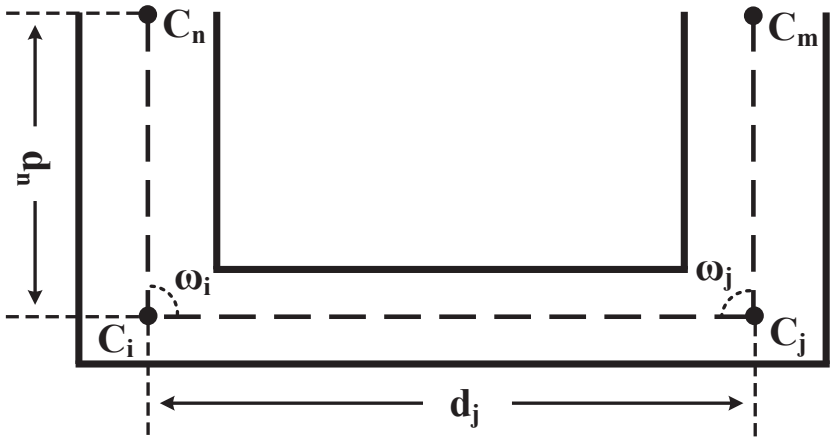


Fig. 3.9: The model of hallway and corner.

in Figure 3.9). Thus k corners have a set of cross points $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$. $\mathbf{D} \in \mathbb{R}$ and $\mathbf{w} \in [-\pi, \pi)$ are the set of the distances of hallways and angles between them. To construct the fine-grained hallway topology, our system calculates the configuration $< \mathbf{D}, \mathbf{w} >$ of hallways, where $\mathbf{D} = (d_1, d_2, \ldots, d_n)$, $\mathbf{w} = (\omega_1, \omega_2, \ldots, \omega_m)$.

Measuring and calibrating the distance D: In this part, we propose a new method to calculate the distance of a hallway through the accelerometer and the depth camera. When a user traverses the hallway, he needs to stop and capture images of rooms distributed along the hallway by Facility Photographing Manner (FPM). This gives us a chance to transfer the problem of calculating the distance d of hallway to the problem of calculating the distance p between two photographing positions as follows:

$$d = \sum_{i=1}^{n} p_i, \tag{3.2}$$

where p_i denotes the distance of two adjacent photographing positions.

Depth data: With the rise of Augmented Reality (AR) technique and the improvement of smartphone computing power, many new sensors have been equipped,

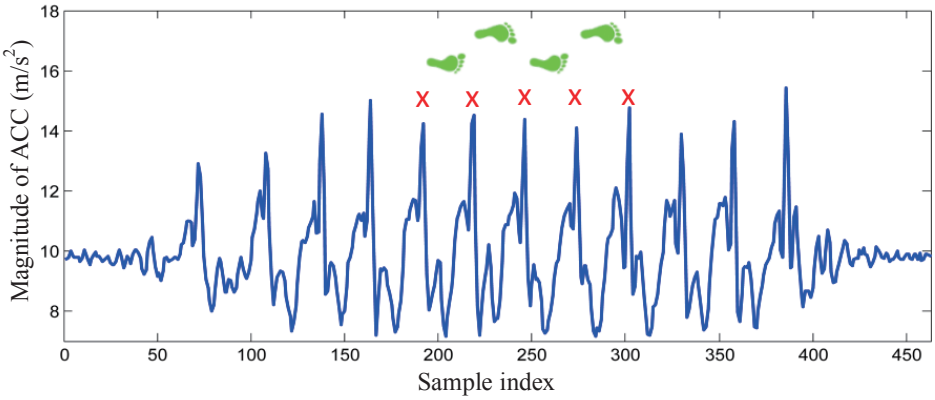


Fig. 3.10: Measuring the distance through counting the steps from the acceleration data.

such as time-of-flight (TOF) depth camera. The TOF depth camera can obtain the distance between the camera and the subject for each point in the image through measuring the time-of-flight of a laser light. Thus, similar with the normal RGB camera, a 3D depth image can be captured by the TOF depth camera with the depth of each point known. Because of the internal property of TOF depth camera, the measured distance can achieve a high accuracy in centimetre level.

We measure the distance between two photographing positions through two manners: accelerometer and depth camera. First, between two adjacent photographing positions, the steps are detected and counted from the acceleration data [Wang et al., 2012, Wu et al., 2015b], as shown in Figure 3.10. The distance is calculated by multiplying the step count with the step size which could be estimated from the user's weight and height. Second, when the user captures an image in the corner or turns to face the hallway after photographing a room, a 3D depth image of the hallway will be captured by the depth camera. Through the position of the recognized room in the RGB image, the room position can be projected into the depth image. Thus, after

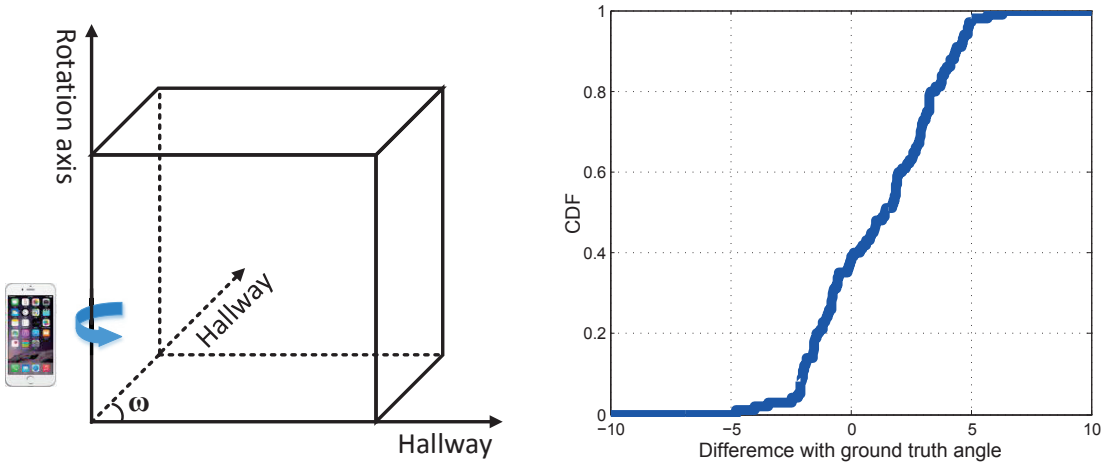
projecting the 3D depth image to the horizontal plane, we can obtain the distance between two adjacent photographing positions from the depth image.

The distance between two photographing positions is calibrated by fusing two measurement results. In our system, an effective particle filter [Rai et al., 2012] is designed to calibrate the measured distances. After activating the data collection, PlanSketcher generates particles spread from the first photographing position. Each particle represents a possible distance between two photographing positions and is updated according to acceleration data. To compensate the noise in step length measurement, a zero mean gaussian noise is added to each particle's step length. The probability density function for particle's step length is:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\|x-\mu\|^2}{2\sigma^2}}, \tag{3.3}$$

where μ is measured distance from acceleration data, σ is the standard deviation. Then, each particle is weighted by the measurement of depth data. PlanSketcher gives greater weights to the more likely particles and the less likely particles are gradually filtered out. The centroid of weighted particles approximates the actual distance between two photographing positions. The weighted particles are resampled after weight normalization. Such calibration process is repeated until the user finishes data collection.

Particle weighting: Next, we describe how PlanSketcher weights each particle according to the observed depth measurement. With the distance measured from projected depth data, a Euclidian difference Δd_i can be obtained for each particle i. When particles need to be updated, their weights are set according to their corresponding Euclidian difference via a Gaussian kernel. In particular, for the i-th



(a) Angle measurement.

(b) Measurement error for 90° corner.

Fig. 3.11: Measuring the angle between two crossed hallways.

particle, its weight is set as follows:

$$weight_i = e^{-\frac{\Delta d_i}{k}}, \tag{3.4}$$

where Δd_i is the Euclidian difference and k is a tunable parameter.

Moreover, if the hallway is traversed repetitively, we use the average of multiple estimations to be the distance.

Measuring and calibrating the angle ω: To acquire the spatial relation of hallways, the angle ω between two crossed hallways is measured and calibrated through the photographing manner and depth data.

First, the angle ω is measured by integrating angular velocities captured by the gyroscope. As the prescribed Corner Photographing Manner (CPM), the user holds the smartphone perpendicularly to the earth surface to photograph in the direction of a hallway, and then turns to another direction. So the rotation angle represents the angle between the two hallways as illustrated in Figure 3.11 (a).

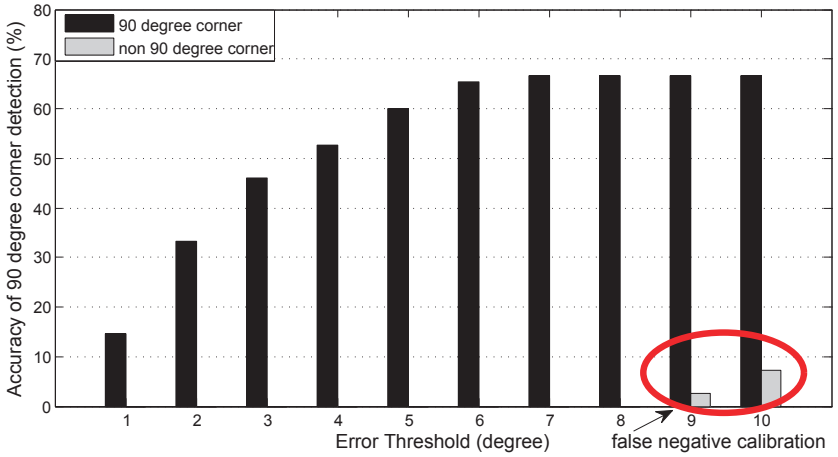


Fig. 3.12: The tradeoff between error threshold and accuracy of 90° corner detection.

Second, after measuring the angle, out system calibrates the measurement value to the truth value for the 90° corner. The idea comes from the observation that most of the corners in the building are 90° and there is a distinct gap between 90° corner and non 90° corner. Figure 3.11 (b) shows the cumulative distribution function (CDF) of angular measurement error of 90° corner. The measurement error is minor, which is benefited from the prescribed manner, with a fluctuation range from $-4.75°$ to $6.3°$. We define the absolute error $E_i \in [0, 2\pi)$ of the corner C_i as:

$$E_i = |M_i - T_i| \tag{3.5}$$

where M_i and T_i denote the measurement value and truth value, respectively. In PlanSketcher, we set a threshold for E to calibrate the angle measurement of 90° corner. The threshold is equal to 5° in our system, indicating that the measurements between $(90° - 5°)$ and $(90° + 5°)$ need to be calibrated to 90° . Of course, rather loose threshold improves the measurement accuracy of 90° corner, but also introduces false negative calibration. The tradeoff is shown in Figure 3.12. We observe that 5° is a reasonable cut-off point, balancing the quantity and accuracy of 90° corner. In fact,

to the best of our knowledge, PlanSketcher is the first to calibrate the measurement of angle to the truth value.

Moreover, considering some corners are not 90°, we calibrate the measured angle by the 3D depth image. After projecting the 3D depth image to the horizontal plane, we use the line detection algorithm [Lee et al., 2009] to extract the outer contour of the hallway. By detecting the cross point of the outer contour, the angle of the corner can be obtained from the 3D depth image. Similar with the hallway distance calibration, we also calibrate the corner angle based on the particle filter. The weight of each particle is decided through the measured angle from the 3D depth data.

3.4.2 Constructing Non-traversed Hallways

When the hallways are interlaced in the building, it is effort-intensive and time-consuming for the user to traverse every hallway. If the user traverses the hallway without repetition, a fraction of hallways could be constructed with the above method. Thus, the traversed hallways construction method is necessary but not sufficient to construct the integrated hallway topology.

To construct the hallways which the user has not traversed, we design a novel non-traversed hallways construction method by comparing *the sequences f of room names* in various fingerprints F of corners. The method is motivated by the observation that the sequences of room names (f_i and f_j) are reserve if they are detected from the photos captured at the two corners of the hallway (C_i and C_j). As illustrated in Figure 3.13, although the user does not traverse the hallway $\overline{C_iC_j}$, the photos are captured in each direction of the two corners as the prescribed corner photographing manner (CPM). And two reverse sequences of room names (such as $[LACOSTE, GEOX]$ and $[GEOX, LACOSTE]$) could be detected from the photos, as shown in Figure

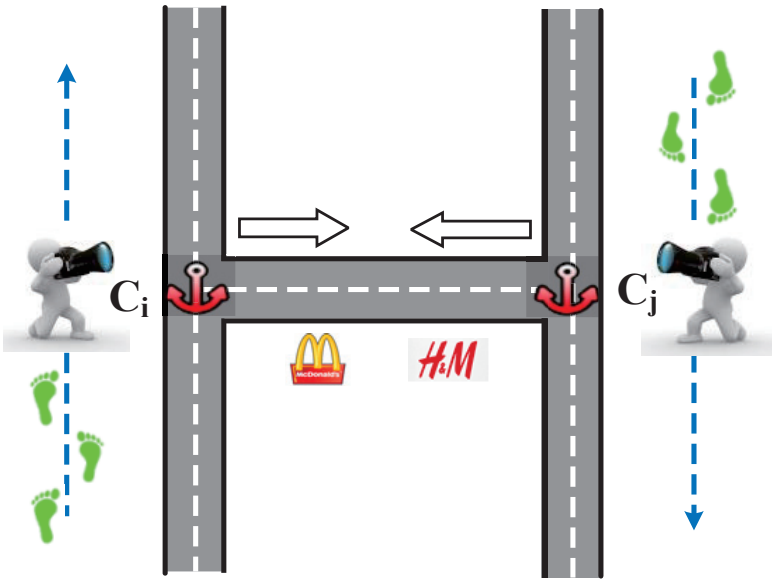


Fig. 3.13: Constructing a non-traversed hallway through image recognition.

3.14. (Photos have been zoomed up and clipped to be more clear.)

The algorithm inputs are the adjacent corner pair which are selected from the previous detected corners and their corresponding images set pair. We assume that for each adjacent corner pair there is an indicator $\Lambda \in \{0, 1\}$ telling whether there exists a hallway between them. Suppose t corner pairs are selected from the previous detected corners. First, for each corner pair (C_i, C_j), the fingerprints of two corners

Fig. 3.14: Two reverse sequences of room names are detected from the photos captured at the corner A and B. (Photos have been zoomed up and clipped to be more clear.)

F_i and F_j are calculated through object detection from the images as the above definition Equation (3.1). Second, we compare every row of room names f_p^i and f_q^j in the fingerprints F_i and F_j. If two reverse sequences are observed, the system infers that the two corners belong to the same hallway and the hallway is constructed to connect them as a linear segment. The whole process is described in Algorithm 1. Thus, PlanSketcher provides an opportunity for the user to construct the non-traversed hallway utilizing the images captured in the corner.

Algorithm 1: Non-traversed Hallway Construction

Input:
adjacent corner pair (C_i, C_j);
corresponding images set pair $(\mathbf{I}_i, \mathbf{I}_j)$;
Output:
hallway indicator vector Λ;

```
for all (C_i, C_j) do
    initialize Λ with 0;
    calculate fingerprint pair (F_i, F_j);
    for all (p, q) do
        if f_p^i == reverse (f_q^j) then
            Λ ← 1;
            break;
```

3.5 Labelled Room Construction

After obtaining the topology of hallways, we construct a relatively complete indoor floor plan with detailed shape and room information by: 1) room construction, and 2) facility labeling. We illustrate these two steps as follows.

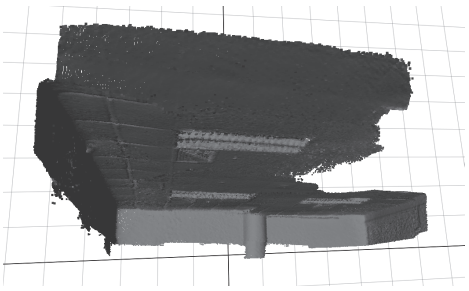
3.5.1 Room Construction

The shape of room is a crucial component in indoor floor plan. Most existing works [Alzantot and Youssef, 2012, Gao et al., 2014, Guo et al., 2014] on room shape construction need users to traverse in the room to collect various data (e.g, inertial sensing data, wireless data or images). However, these approaches are labor-intensive and the accuracy of constructed room shape is still limited. Thus, we propose a novel method to construct room shape from depth data with high accuracy but without traversing in the room.

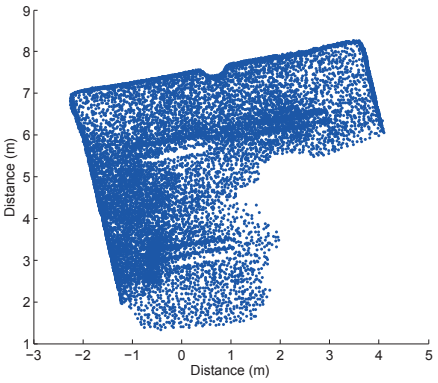
In our system, we leverage the TOF depth camera equipped in smartphone to construct the room shape. After the user takes a photo of a room using FPM, he the captures a 3D depth image of the scene in the room. Considering the problem of obstacles in the room, we ask the user to capture a depth image including the ceiling and the connecting line of wall segments. The insight of this manner is that the ceiling can effectively represent the room shape with less obstacles. Figure 3.15 (a) shows the scene of a room and Figure 3.15 (b) is the 3D depth image captured by smartphone. When the user captures a 3D depth image, the pose of smartphone can be derived from sensor readings, i.e. yaw, pitch and roll. Then we obtain a rough room shape through projecting the 3D depth image to the horizontal plane, as shown in Figure 3.15 (c).

To construct completely room shape, we model the room by major geometric vertices and edges. First, we use the line detection algorithm [Lee et al., 2009] to extract the outer contour of the rough room shape (shown in Figure 3.15 (d)) and infer the line segments of the room (shown in Figure 3.15 (e)). Next, we extend the line segments to find the major geometric vertices in the room, as the two blue

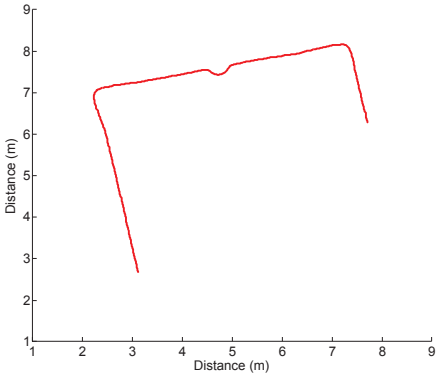
(a) The scene of a room.



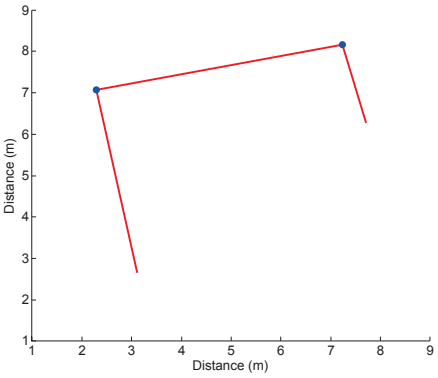
(b) 3D depth image captured by smartphone.



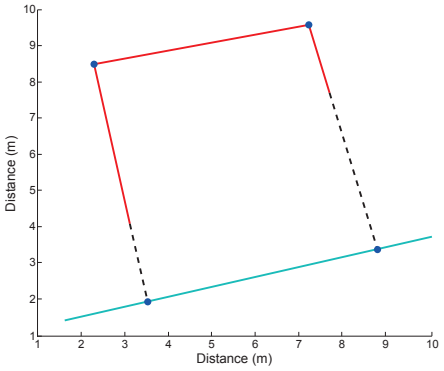
(c) Projection in the horizontal plane.



(d) Extracting the outer contour of the room.



(e) Inferring line segments and major geometric vertices of the room.



(f) Constructing completely room shape.

Fig. 3.15: Illustration of room shape construction.

points shown in Figure 3.15 (e). Then we transfer the room shape constructed in the local coordinate system to the global coordinate system of the indoor floor plan, according to the position of photographing. Moreover, to derive the connecting point between the room and the hallway, we also extend the line segments in two sides to produce the two cross points. Figure 3.15 (f) gives an example, where the blue line is the hallway. Especially, for some irregular room, simply capturing one depth image may not cover all boundaries of walls in the room. The user can make a scanning of boundaries of the ceiling to construct the room.

Different from previous works, the room construction method we proposed do not need user to traverse in the room and can achieve a high accuracy. This non-traversed method can save much labor, time and energy consumption in the smartphone. And because of the internal fine-grained property of laser measurement in the TOF depth camera, the accuracy of constructed room shape is improved.

3.5.2 Facility Labeling

In this subsection, we calculate the entrance positions and orientations of facilities and label them in the constructed hallway plans and rooms. The names and categories of facilities are extremely valuable information for user's reference. It is difficult for users to find their destinations and know the surroundings with merely the topology of hallways and rooms. Thus the complete indoor floor plans should provide the names, categories and positions of facilities.

The positions of elevators, escalators, stairs (viewed as the real landmarks) and restrooms are directly derived from the recorded inertial sensing data and the positions of photographing. The positions and orientations of rooms are estimated from the facility photographing manner (FPM).

The proposed FPM provides a cue to estimate the positions of entrances $\mathbf{E} = (E_1, E_2, \ldots, E_n)$. From our practical observations, we find that the logos are usually placed on the top of or beside the entrances. Because the user takes photos in front of rooms, we set the photograph's locations as the positions of entrances, which are derived from the acceleration data. The orientations of facilities are inferred from rotation angles when the user photographs. The rotation angle is calibrated as the hallway angle measurement through the threshold-based approach. After calculating the entrance positions and orientations of facilities, PlanSketcher labels the facility icons or names in the hallway plans. The spatial relations of facilities are ensured from the FPM. Thus, it is more accurate to locate facilities than the conventional trajectory-based approaches.

3.6 Implementations and Evaluation

In this section, we present the evaluation of key functional components of PlanSketcher. We evaluate PlanSketcher in three representative indoor environments for a better understanding of the effectiveness and limitation of system.

3.6.1 Experimental Methodology

We implement PlanSketcher on the Android platform (version 4.4 KitKat) based on JAVA. The OpenCV library (version 3.0) is adopted to implement the image recognition. In particular, a training data (images) set is built in advance to conduct the image recognition. The training images are collected from two parts, which contains 2,000 images about 100 categories of facilities, alphabet and numbers. The first part images (300 out of 2000 samples) are collected via photographing the facilities from

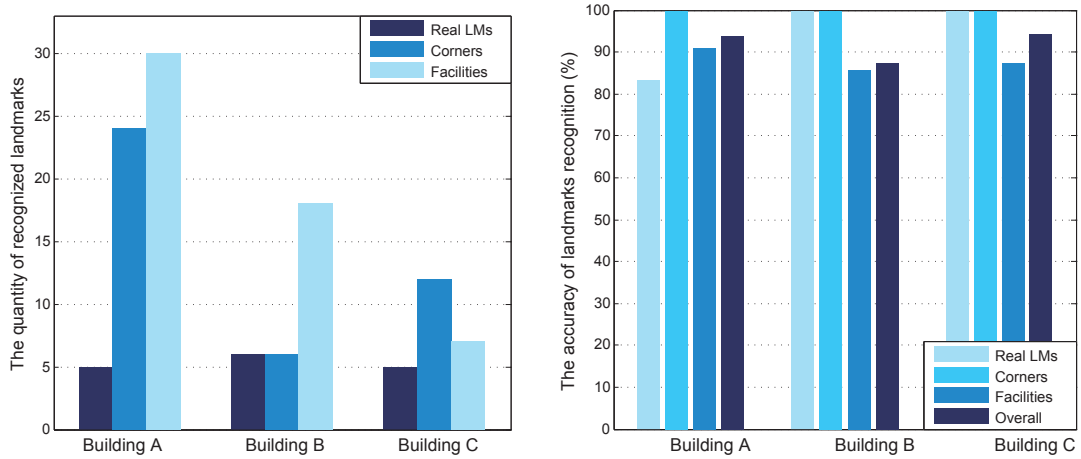
various viewpoints in the real situation (different from the test buildings in the followed section). The second part images (1700 out of 2000 samples) are downloaded from the internet to enable the generality of the images. Then for every image, we first extract the histogram of oriented gradients (HOG) [Dalal and Triggs, 2005] and scale-invariant feature transform (SIFT) features [Lowe, 2004] to represent the local image information, then apply the Locality-constrained Linear Coding (LLC) method [Wang et al., 2010] to further encode the local features into final image histograms. The LLC utilizes the locality constraints to project each descriptor into its local-coordinate system and captures the correlations between similar local features by sharing the visual words, thus it could give better detection results compared with the traditional bag of visual words methods [Fei-Fei and Perona, 2005]. For the image classification, instead of utilizing the traditional support vector machine (SVM) in the OpenCV [Ope], we apply the Multiple Kernel Boosting (MKB) [Yang et al., 2011] to classify the images. MKB is a boosted Multiple Kernel Learning (MKL) method, which combines several SVMs of different kernels, thus it could provide better classification performance. To obtain the absolute walking direction in the indoor floor plan construction, we leverage the loss of GPS signal to detect whether a user enters a building. The absolute walking direction can be obtained at the building entrance. After the user enters the building, the walking direction can be derived from inertial sensing data.

3.6.2 Performance Evaluation

We test PlanSketcher on a variety of Android mobile devices (Lenovo Phab2 Pro, Google Nexus 5, Samsung Galaxy S2 and Samsung Galaxy Note 3). Especially, we use Lenovo Phab2 Pro to gather inertial sensing data, WiFi RSSI values, depth data

and images in three large complex indoor environments: one story of a $145m \times 40m$ shopping mall (Building A), one story of a $100m \times 40m$ university building (Building B) and one story of a $40m \times 40m$ exhibition center (Building C). Because other three mobile devices have not been equipped with depth camera, we test their energy consumption of collecting inertial sensing data, WiFi RSSI values and images. The lighting conditions during open hours allow user to capture bright images. The user holds the smartphone in a perpendicular angle to the ground to collect trajectories and images in the buildings. In each environment, we take 38, 16 and 31 photos of different corners in Corner Photographing Manner and collect 33, 27 and 17 photos of different facilities in Facility Photographing Manner. We also gather 12, 6 and 8 user trajectories along the hallways in each building.

Performance of Landmark Recognition. We evaluate the quantity of recognized landmarks and the accuracy of landmarks recognition. Figure 3.16 (a) shows the quantity of landmarks recognized from the sensing data and images in each building. For the building A, the composition of the landmarks is: 5 real landmarks, 24 corners and 30 facilities. For the building B, the composition is: 6 real landmarks, 6 corners and 17 facilities. For the building C, the composition is: 5 real landmarks, 12 corners and 7 facilities. Because PlanSketcher recognizes various landmarks, the well-known cumulative error is limited to a low level, which improves the accuracy of whole system. Figure 3.16 (b) shows the accuracy of landmarks recognition for all categories of landmarks. By the proposed integrated features, most of the real landmarks are recognized (only 1 missed in building A). The reason for the undetected real landmark (an escalator) is the low magnitude of RSSI (less than -80dBm), which cannot help the acceleration readings to present a significant feature. All corners



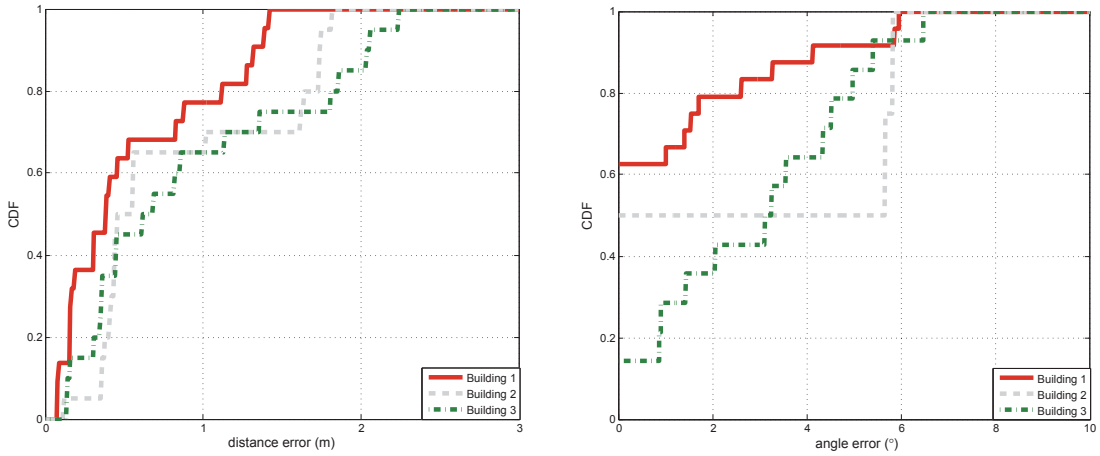
(a) The quantity of recognized landmarks.

(b) The accuracy of landmarks recognition.

Fig. 3.16: The Performance of Landmark Recognition.

are detected and the recognition rate of facilities is more than 85%. Although some landmarks are missed, the recognition still achieves a high accuracy with the overall rate: 93.65%, 85.29% and 94.22% in three buildings.

Performance of Hallway Construction. Precision of the constructed hallway is a critical criteria of PlanSketcher. We use the distance errors and angle errors of conducted hallways (i.e., the difference between the calculation values and the ground truth measurements) to evaluate the performance. As shown in Figure 3.17 (a), PlanSketcher generates precise distances of hallways. The constructed hallways achieve average accuracy of $0.51m$, $0.69m$ and $0.83m$ and 90th percentile accuracy of $1.32m$, $1.72m$ and $2.02m$ in each building. Because abundant landmarks are recognized in the building and the accurate depth data, the accumulation error of distance is limited to a low level. As shown in Figure 3.17 (b), PlanSketcher produces accurate angles of hallways. PlanSketcher yields the angles of hallways, with average accuracy of 1.15°, 2.87° and 2.92° and 90th percentile accuracy of 4.14°, 5.84° and



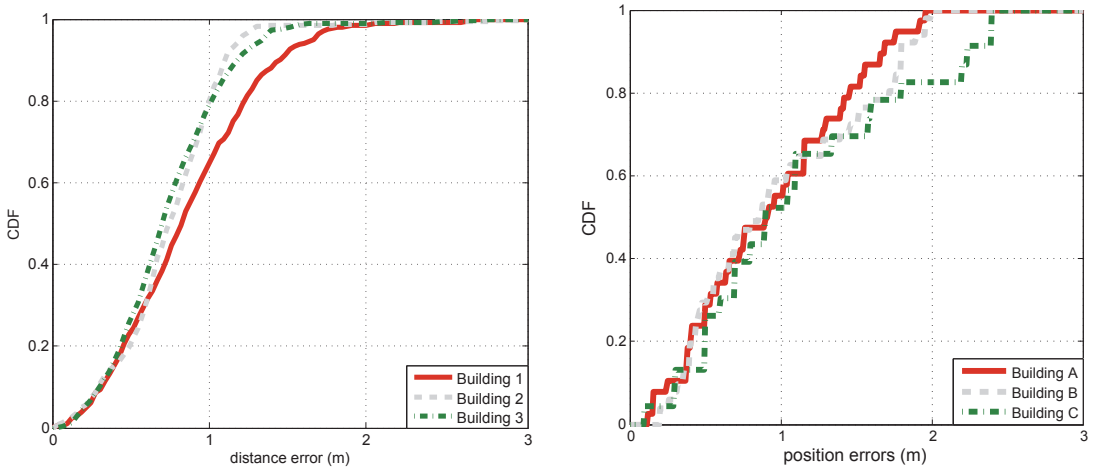
(a) Distance errors of constructed hallways.

(b) Angle errors of constructed hallways.

Fig. 3.17: The Performance of Hallway Construction.

5.42° in each building. The high accuracy is benefitted from the stable performance of Corner Photographing Manner (CPM) and hallway angle calibration. Most angle measurements of corners have been calibrated to the truth values.

Performance of Labelled Room Construction. We evaluate the performance of labelled room construction using the distance errors of constructed rooms and position errors of facility labeling. Figure 3.18 (a) shows the distance errors of constructed room edges. The constructed room edges achieve average accuracy of $0.84m$, $0.61m$ and $0.72m$ and 90th percentile accuracy of $1.37m$, $1.07m$ and $1.21m$ in each building. Because of the internal property of high accuracy in TOF depth camera, the distance error of edge is limited to a low level. Figure 3.18 (b) shows the position errors of room entrances where are used to label facilities. The average errors of room entrances are $0.94m$, $0.98m$, $1.1m$ and 90th percentile accuracy of $1.68m$, $1.79m$, $2.23m$ in each building. The high accuracy of facility positions is achieved, because abundant recognized landmarks in the buildings help to calibrate the measured distances.



(a) Distance errors of constructed rooms.

(b) Position errors of facility labeling.

Fig. 3.18: The Performance of Labelled room Construction.

Energy Consumption. PlanSketcher employs inertial sensors, WiFi, camera as well as depth camera to collect various data for indoor floor plans construction. Thus, given the energy bottleneck of smartphones, the energy consumption issues should be taken into consideration and discussion for practical use. We measure the energy consumption on various models of smartphones utilizing the Monsoon power monitor [Mon]. The power monitor directly supplies power to the smartphone and accurately tracks the current, voltage and power. To precisely measure the energy consumption of PlanSketcher, all background services and applications are turned off. The WiFi module is turned on and the screen brightness is set to auto-adjustment mode. All the sensor modules and WiFi module are sampled and processed in real time. As the smartphone has to be wired with the power monitor for measurement, we keep the smartphone stationary or moving in limited space. We synthetically trigger image capture in PlanSketcher every 5s to 20s.

Figure 3.19 shows the working currents measured on the Lenovo Phab2 Pro with a

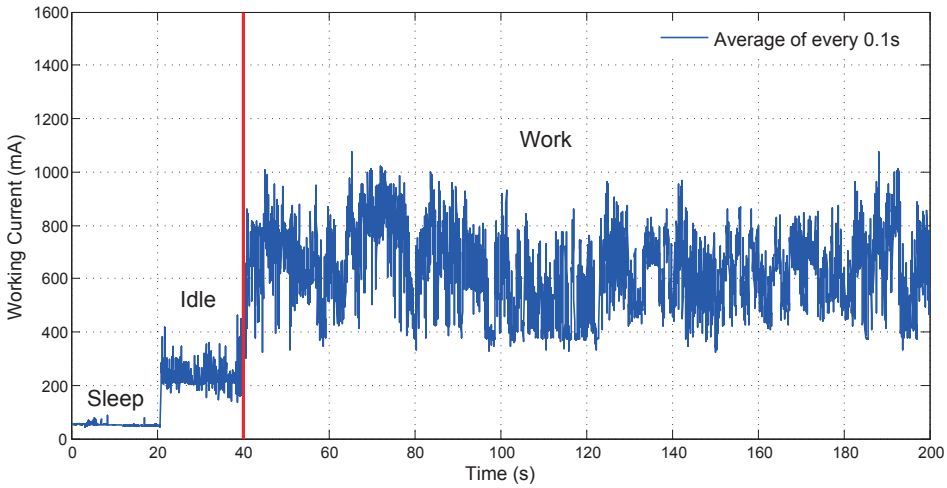


Fig. 3.19: Energy Consumption.

4050mAh battery in different working modes as an example. The current is sampled at 5KHz utilizing the power monitor and averaged over every window of 0.1s. During the evaluation, the smartphone is in sleep mode during the period from 0s to 20s. We wake up and unlock the smartphone during the period from 20s to 40s. And then we launch PlanSketcher at around 40s and start to collect various sensing data and images. The data collection is finished at 200s. We repeat the experiments 10 times and characterize the power draws in different modes in Table 3.1. The work mode of PlanSketcher draws power at around 640.83mA and the expected battery life is 6.32h for the Lenovo Phab2 Pro.

We also measure the expected battery life of the Google Nexus 5 with a 2300mAh battery, the Samsung Galaxy S2 with a 1800mAh battery and Samsung Galaxy Note 3 with a 3200mAh battery. Because these mobile devices have not been equipped with depth camera, we test their energy consumption of collecting inertial sensing datap, WiFi RSSI values and images. The measurement results are presented in Table 3.2. For the Google Nexus 5, it can continuously work for about 6.32h. The

Table 3.1: The Energy Consumption on Lenovo Phab2 Pro

Mode	**Period**	**Power**	**Current**	**Battery Life**
Sleep	$0s \sim 20s$	$190.90mW$	$51.18mA$	$79.13h$
Idle	$20s \sim 40s$	$884.12mW$	$237.03mA$	$17.09h$
Work	$40s \sim 200s$	$2390.30mW$	$640.83mA$	$6.32h$

Table 3.2: Battery Life Measurements in Different Models of Smartphones

Energy	**Phab2 Pro**	**Nexus 5**	**Galaxy S2**	**Note 3**
Battery Capacity	$4050mAh$	$2300mAh$	$1800mAh$	$3200mAh$
Battery Life (Work)	$6.32h$	$4.47h$	$3.11h$	$4.91h$

expected battery life of Google Nexus 5 is 4.47h in work mode. Powered by a small battery, the expected life of Samsung Galaxy S2 is around 3.11h in work mode. The expected battery life of Samsung Galaxy Note 3 is 4.91h in work mode. The expected battery life time is longer for the Lenovo Phab2 Pro compared with the others, mainly because of its larger battery.

The current version of PlanSketcher can be further optimized for energy efficiency [Liu et al., 2013, 2015, Shu et al., 2013, Zhang et al., 2015]. PlanSketcher may reduce the image capture quality to a discernible level to save energy and benefit from the energy efficient mobile vision techniques [LiKamWa et al., 2013]. When the user traverses in the indoor environment, PlanSketcher may dynamically adjust WiFi scanning rate to reduce energy consumption. PlanSketcher can also benefit from energy efficient coprocessor architectures for sensor fusion as well [Shen et al., 2014].

Performance Comparison. To further evaluate our proposed system, we implement two systems (Jigsaw [Gao et al., 2014] and Shopprofiler [Guo et al., 2014]), and compare the construction performance with them. Jigsaw constructs indoor hallway and room topologies from the inertial sensing data and images. Shopprofiler utilizes the inertial sensing data, acoustic data and WiFi access information to construct indoor floor plans and recognize shops in the buildings. Both of the systems use the crowdsourcing approach to collect data. In the experiment, we recruited 10 volunteers to collect total 100 user traces (including acceleration and gyroscope data) and 250 photos in each building to implement Jigsaw. For Shopprofiler, we also recruited 10 volunteers to collect total 100 user traces (including acceleration data, gyroscope data, acoustic data and WiFi access information) in each building. As shown in Table 3.3, PlanSketcher outperforms the other two schemes. PlanSketcher achieves a higher accuracy ($1th$ and $2th$ rows in Table 3.3) of indoor floor plans (including hallways and rooms) than the others, because much landmarks are recognized and most of the measured angles are calibrated to the truth values. PlanSketcher can recognize more facilities ($3th$ and $4th$ rows in Table 3.3) based on the superior visual object detection technique instead of inaccurate WiFi information. In addition, we test the energy consumption in smartphone of completing the input data collection in each building for each system. The battery of smartphone is recorded when data collection is started and finished. Our system consumes less energy in smartphone ($5th$ row in Table 3.3). This is because: 1) our system is a non-crowdsourcing-based approach which spends less time collecting less amount of data, and 2) the user is unnecessary to traverse all hallways and rooms. We notice that Shopprofiler fails to recognize most of the facilities in the buildings. This is because a fraction of stores or rooms

Table 3.3: The Performance Comparison

Category	Jigsaw	Shopprofiler	PlanSketcher
Position error	$2m \sim 3m$	$2m \sim 5m$	$1m \sim 2.5m$
Orientation error	$6° \sim 9°$	$8° \sim 15°$	$4° \sim 6°$
Facility quantity	None	$9 \sim 25$	$20 \sim 40$
Recognition rate	None	$30\% \sim 60\%$	$85\% \sim 95\%$
Energy Consumption	$843mAh \sim 1278mAh$	$411mAh \sim 634mAh$	$224mAh \sim 358mAh$

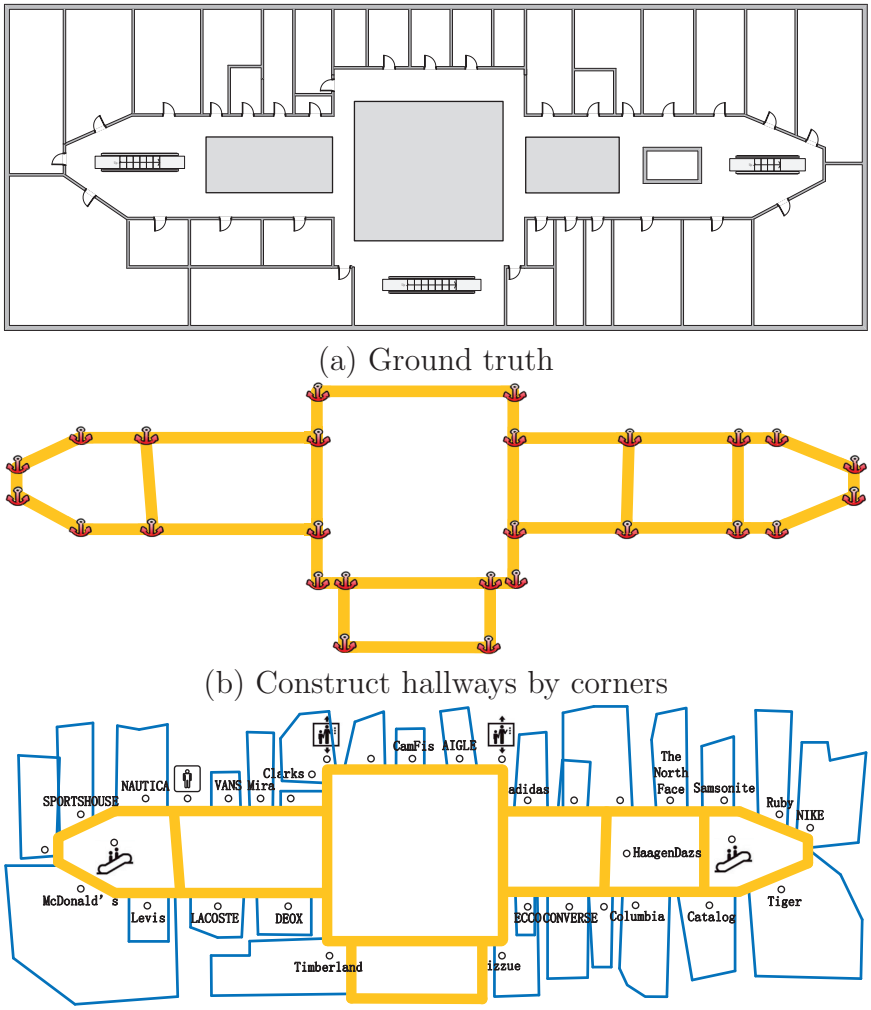
provide their own WiFi access while the others are provided by the buildings which results in the same WiFi SSID. The above results demonstrate that PlanSketcher can construct the facility-labelled and highly fine-grained indoor floor plans with little energy consumption in smartphone.

Floor Plan Performance. Figure 3.20 shows the ground truth and the process of labelled indoor floor plan construction in the shopping mall. The shadow areas represent some inaccessible areas. In Figure 3.20 (b), the anchor icons highlight all detected corners and the lines show constructed hallways. When two corners are detected belonging to the same hallway, they are connected with a straight line to form a hallway. In Figure 3.20 (c), the blue lines indicate the constructed room shape. Especially, many room shapes are different from the ground truth, because these shop owners restructure the room space to generate extra functional zones (such as storeroom). Facilities are recognized and labelled in the floor plan, which involve elevators, escalators, restrooms and stores etc. The circles indicate the positions of facility entries. Because the training samples may not be collected perfectly sufficient,

a small quantity of facilities (1 escalator and 3 stores) are not recognized and labelled. Compared with the ground truth, the spatial relations of hallways and facilities are all correct in the constructed floor plans. Moreover, Figure 3.21 and Figure 3.22 show the constructed indoor floor plans in the university building and the exhibition center, respectively. Compared with the room shapes constructed in shopping mall, the room shapes are more consistent with the architectural drawings in these two buildings. This is because the room space is not restructured in these buildings, which demonstrates the effectiveness of our proposed methods. The spatial relations of hallways are correct and more than 85% facilities are recognized and labelled successfully.

3.7 Chapter Summary

In this chapter, we utilize machine learning techniques to propose PlanSketcher, a system that enables a user to construct indoor floor plans by himself. Compared with previous works, PlanSketcher can construct fine-grained and facility-labelled indoor floor plan with less energy consumption. To realize this system, various sensing data are collected from smartphone and novel landmarks recognition approaches are presented. Then novel hallways construction algorithms are proposed to construct traverse-independent hallway topologies. With the object detection technique, PlanSketcher also constructs the room shape and labels recognized facilities in their corresponding positions. We implement PlanSketcher and conduct abundant experiments. The evaluation results illustrate that PlanSketcher outperforms the state-of-the-art methods by showing smaller position and orientation error, more recognized facilities and less energy consumption.

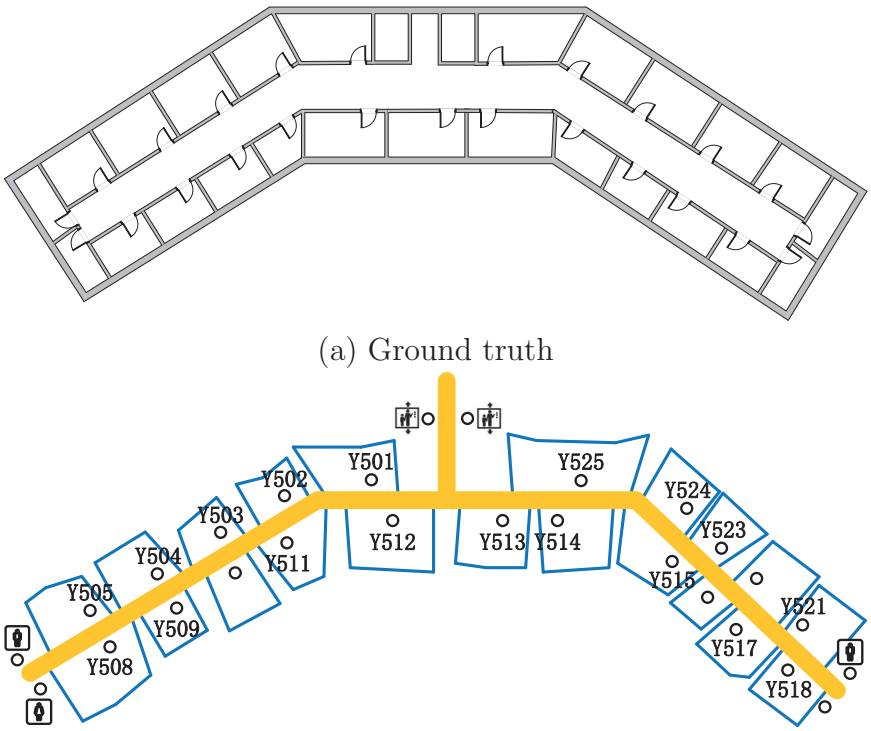


(a) Ground truth

(b) Construct hallways by corners

(c) Construct indoor floor plan with labels

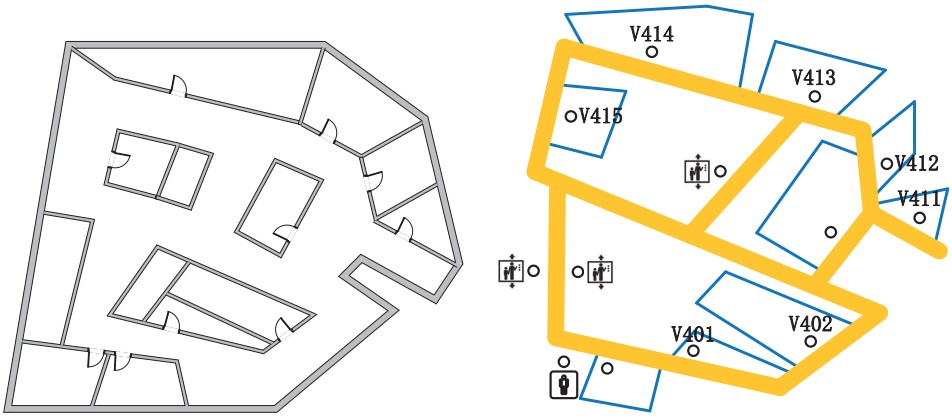
Fig. 3.20: The Ground Truth and Constructed Indoor Floor Plan with Labels in the Shopping Mall.



(a) Ground truth

(b) Construct indoor floor plan with labels

Fig. 3.21: The Ground Truth and Constructed Indoor Floor Plan with Labels in the University Building.



(a) Ground truth

(b) Construct indoor floor plan with labels

Fig. 3.22: The Ground Truth and Constructed Indoor Floor Plan with Labels in the Exhibition Center.

Chapter 4

Updating Digital Maps via Mobile Crowdsensing

Accurate digital maps play a crucial role in various location-based services and applications. However, store information is usually missing or outdated in current maps. In this chapter, we propose CrowdGIS, an automatic store self-updating system for digital maps that leverages street views and sensing data crowdsourced from mobile users. We first develop a new weighted artificial neural network to learn the underlying relationship between estimated positions and real positions to localize user's shooting positions. Then, a novel text detection method is designed by considering two valuable features, including the color and texture information of letters. In this way, we can recognize complete store name instead of individual letters as in the previous study. Furthermore, we transfer the shooting position to the location of recognized stores in the map. Finally, CrowdGIS considers three updating categories (replacing, adding, and deleting) to update changed stores in the map based on the kernel density estimate model. We implement CrowdGIS and conduct extensive experiments in a real outdoor region for 1 month. The evaluation results demonstrate that CrowdGIS effectively accommodates store variations and updates stores

to maintain an up-to-date map with high accuracy.

4.1 Overview

The proliferation of mobile computing has prompted the development of map construction techniques based on mobile crowdsourcing. An accurate map with abundant store information can provide users efficient location-based services, including localization, navigation and information sharing. However, stores may be changed and update information may not be available in current maps. Based on real investigations, current digital maps still lack a large amount of store information. Moreover, replaced stores and nonexistent stores cannot be timely updated in the map neither. For example, in some large cities around the world such as Hong Kong, Tokyo and New York, we found plenty of stores are not labelled in the digital map and changed very frequently. Existing inaccurate and out-of-date maps may misguide users and even bring dangers. Therefore, it is crucial to automatically update stores in maps (i.e. replace old stores, add newly-built stores and delete nonexistent stores) to provide better location-based services.

Recently, many works have been devoted to reconstruct and update maps. These works can be classified into three types. The first type is to reconstruct maps based on simultaneous localization and mapping (SLAM) [Bailey and Durrant-Whyte, 2006, Civera et al., 2015]. ORB-SLAM2 [Mur-Artal and Tardós, 2017] utilizes monocular, stereo, and RGB-D sensors to perform relocalization, loop closing, and reuse its map in real time on standard CPUs. Authors of [Mohanarajah et al., 2015] present an architecture, protocol, and parallel algorithms for collaborative 3D mapping in the cloud with low-cost robots. The robots run a dense visual odometry algorithm on a

smartphone-class processor. The second type updates maps through a manual survey. The state-of-the-art Google map leverages crowdsourcing for map updating, where user-submitted changes are integrated into their map after a manual review. The third type is to automatically update maps. CrowdAtlas [Wang et al., 2013] automate road updating in a map based on people's travels, either individually or crowdsourced. It uses a mobile navigation app to detect significant portions of GPS traces that do not conform to the existing map. Roads are updated in the map when sufficient traces are collected. AcMu [Wu et al., 2015a] updates WiFi Received Signal Strength (RSS) of each position in a map for wireless indoor localization. By accurately pinpointing mobile devices, the system can collect real-time RSS samples when devices are static. With these reference data, the system updates the complete radio map by learning an underlying relationship of RSS dependency between different locations.

While existing studies have tried to explore the possibility of updating map, accurate store update in a digital map deserves more attention. First, SLAM-based approaches mainly focus on reconstructing maps, which cannot be directly utilized to find changes and update maps. And these methods often need extra devices, such as depth camera, robot and vehicle. Second, updating stores through manual reviews as Google is both effort-intensive and time-consuming [Chen et al., 2015b]. Even though the store owner can actively apply to update his store, such information is not sufficient and changed stores may not be updated in time, especially for nonexistent stores. Third, previous works can update specific components in maps. For example, crowdAtlas [Wang et al., 2013] focuses on updating changed roads in maps from GPS traces collected via crowdsourcing. AcMu [Wu et al., 2015a] mainly updates WiFi

RSS values of different positions in maps from sensor data. However, in a real situation, besides roads and WiFi RSSs, stores also need to be updated, such as replacing old stores, adding newly-built stores and deleting nonexistent stores. The names and positions of stores in maps are extremely valuable information for user's reference. Third, these studies heavily rely on sensory data. Except sensory information, visual information can preserve more context information for an unknown environment, such as the geometric information, color information, and text information. Visual information based approaches may provide more accurate geometric (shape, coordinates and orientations) information compared with the sensor-only approaches. As a consequence, our further research problem would be: *Can we provide a practical and effective approach to automatically update stores in a digital map through mobile crowdsensing?*

In this chapter, we propose an affirmative answer through the systematic design and implementation of *CrowdGIS*, which enables stores to be automatically replaced, added, and deleted in maps leveraging mobile crowdsourced data. Different from GPS-based schemes, we estimate user's shooting positions from both GPS and images by proposed joint position estimation scheme. Specially, an underlying relationship between estimated positions and real positions is learned through developed new weighted artificial neural network. Then, a novel text detection method is designed by considering two valuable features to recognize complete store name instead of individual letters as previous study. After that, we transfer the shooting position to store position in the map. According to real observations, we further consider three various categories of updating stores: replacing old stores, adding newly-built stores and deleting nonexistent stores. To accurately localize and update changed

stores, position estimation method is proposed based on the kernel density estimate model. CrowdGIS can save extensive manpower and time to effectively update stores in a digital map. When more stores are updated, users can locate their positions more precisely and receive much better location-based services. To the best of our knowledge, our work represents the first attempt to cope with store variations to automatically update stores via mobile crowdsourcing.

The automatical store update requires store localization and recognition from street views taken from smartphones. Thus, implementing such a functional system entails distinct challenges. (1) Localizing shooting positions with high accuracy from street views and sensing data. (2) Precisely recognizing various stores from street views. (3) Accurately localizing the stores recognized from street views in the map. (4) Updating changed stores in the map based on their various categories.

To address the above challenges, we make the following contributions in this chapter:

- We propose the *CrowdGIS* system architecture which leverages mobile crowdsourced data to automatically update stores in a digital map.

- Different from previous GPS-based schemes, we localize user's positions from both GPS and images. An underlying relationship between various estimated positions and the real position is learned to accurately localize user's shooting positions. The average localization accuracy can be improved by about 25%.

- A novel store name recognition method is proposed by considering two valuable features (i.e. the position of text in image and the colour histogram). In this way, we are able to recognize the complete store name instead of individual

letters as previous study. The results show that about 80% store names can be accurately recognized.

- According to real observations, we consider three various categories of updating stores: replacing old stores, adding newly-built stores and deleting nonexistent stores. To accurately localize changed stores, we estimate their positions based on the kernel density estimate model. More than 75% stores can be updated correctly, and an average accuracy of about 8 meters can be achieved.

- In addition, we develop a prototype and conduct extensive evaluations in a real outdoor region for 1 month. The results illustrate that CrowdGIS effectively accommodates store variations and maintains an up-to-date map.

The remainder of this chapter is organized as follows. We first provide the system overview of CrowdGIS in Section 4.2. Then we provide preliminary techniques for our system in Section 4.3. Followed is our system design in Section 4.4. The implementations and evaluation are illustrated in Section 4.5. Finally, we conclude this chapter in Section 4.6.

4.2 System Design

High-performance sensors can provide abundant movement information and street views can offer luxuriant visible and valuable description about surroundings. Accordingly, we propose CrowdGIS system to update stores in the map leveraging street views and sensing data crowdsourced from smartphones. This system consists of four major stages. The CrowdGIS system architecture is sketched in Figure 5.1.

Shooting Position Localization. CrowdGIS utilizes street views and sensing

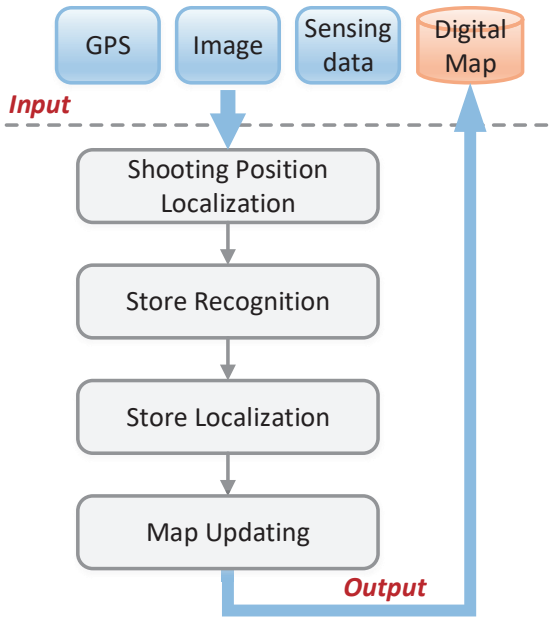


Fig. 4.1: The architecture of CrowdGIS system.

data to jointly estimate the shooting positions of captured street views. For each captured street view, we calculate four shooting position candidates by four various methods (i.e. one GPS method and three image matching methods). Then the shooting position is predicted through the relationship between four shooting position candidates and the real shooting position, which is pre-learned from collected training data.

Store Recognition. Considering two types of stores (with or without logos), we integrate two store recognition approaches to recognizing various store names from street views. For stores with logos, we utilize the object detection technique to recognize them from logos. For stores without logos, we propose a novel store name recognition method to divide and extract store names by considering two effective features (distance and colour).

Store Localization. After recognizing store names, CrowdGIS localizes these

stores in the map based on the captured street views and corresponding shooting positions. We extract the position of store in street view and then transfer it into the global coordinate system of the map through calculating its shooting direction.

Map Updating. With the obtained names and positions of stores from various street views, we propose a map updating method to update stores in the map. Based on observations in real environment, we classify changed stores into three categories (i.e. replacing old stores, adding newly-built stores and deleting nonexistent stores) and then update them with corresponding approach. Specifically, we design a position estimation model to calculate the accurate position of updating store based on kernel density estimate model.

4.3 Preliminaries

In this section, we briefly review some techniques behind our system, and clarify their necessity for our purpose.

Pinhole Camera: We use the classical Pinhole Camera [Sturm, 2014] to model the imaging principle and photograph parameters acquiring. In this model, 3D points in the real world and their projected points in the image plane construct an ideal pinhole camera, where its aperture is described as a point and no lenses are used for focusing light. In this way, we can ignore geometric distortions and unfocused blurring caused by lenses and finite sized apertures. Based on this model, we can acquire the angle of view, the size of photo from the smartphone's camera parameters, which are essential factors for store recognition and localization.

Artificial Neural Network: Artificial Neural Network (ANN) [Sutskever et al., 2014] is a computational multi-layer model based on the structure and functions of

interconnected neurons. ANN is utilized to find relationships between inputs and outputs given finite data samples. The expression of the weighted sum to the k-th neuron in the j-th layer ($j \geq 2$) is given by

$$S_{j,k} = \sum_{i=1}^{N_{j-1}} (\omega_{j-1,i,k} I_{j-1,i}) + b_{j,k} \tag{4.1}$$

where $I_{j-1,i}$ is the information from the i-th neuron in the $(j-1)$-th layer, $b_{j,k}$ is the bias term and N_{j-1} is the number of neurons in the $(j-1)$-th layer. Thus, drawing on the ANN framework, we can find the unknown functions between several estimated positions and real position to accurately localize the shooting position of captured street view.

Manhattan World Assumption: Most man-made scenes follow the Manhattan World Assumption [Vanegas et al., 2012], where Cartesian coordinate system is used as a Manhattan grid. All lines in a photo image are assumed parallel to three directions. Accordingly, we can extract text aligned with the three Manhattan directions from street view. Hence various store names can be recognized through clustering extracted letters and numbers.

Kernel Density Estimate: Kernel Density Estimate (KDE) [Botev et al., 2010] is a data smoothing method used to estimate the probability density function based on a finite data sample set. Let $(x_1, x_2, ..., x_n)$ be an independent and identically distribution samples drawn from some distribution with an unknown density function f. The unknown density function f can be estimated by kernel density estimate (KDE) as following:

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^{n} K_i(\frac{x - x_i}{h}), \tag{4.2}$$

where $K_i(\cdot)$ is the kernel function (a nonnegative function that integrates to one and has mean zero) of the sample x_i, and $h > 0$ is a smoothing parameter called

bandwidth. Since the store position is unknown, the KDE is a powerful tool to estimate the probability distribution of store position from a finite set of candidate positions in the map.

4.4 Store Updating System

In this section, we first illustrate proposed method to localize shooting positions from collected street views and sensing data. Then we present schemes to recognize and localize various stores from captured street views, and further update changed stores in the map.

4.4.1 Shooting Position Localization

In this subsection, we propose a novel scheme to localize the shooting position of street views captured by user's smartphones. Traditional methods just utilize GPS data collected by smartphones to localize shooting position. However, the accuracy of these methods is very limited because of the inherent error in GPS sensor. Instead, street views captured by users also provide extremely valuable information about surroundings, which can be utilized to localize shooting position. With this insight, we propose a novel joint position estimation algorithm to localize the shooting position of street view, combining the GPS data and images. Figure 4.2 gives an example of capturing a street view from shooting position.

Shooting position estimation from GPS. We collect a sequence of GPS samples to estimate shooting position. The GPS data contain two parts: one GPS sample while photographing which is called *start data* denoted as s, and a series of GPS samples when the user moves after photographing which are called *tail data* denoted as

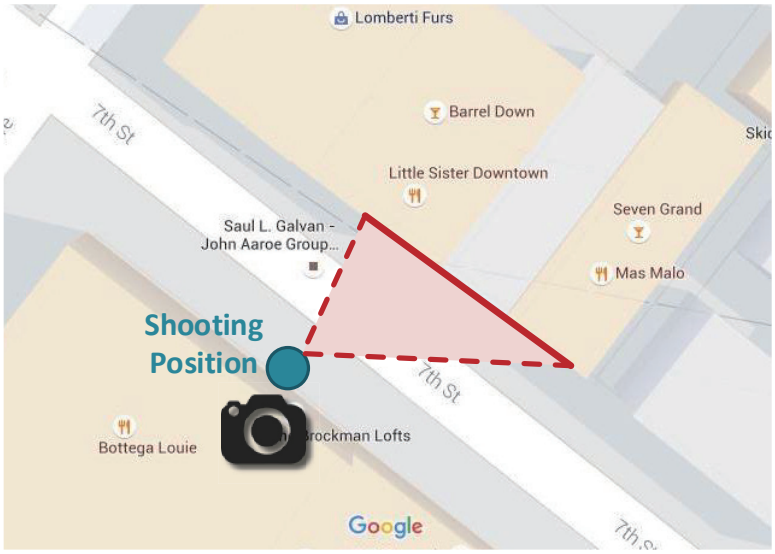


Fig. 4.2: An example of capturing a street view from shooting position.

$\{t_1, t_2, \cdots, t_n\}$. The estimated shooting position from GPS is denoted as $\mathbf{x}_G$. To calculate the position $\mathbf{x}_G$, we utilize a road matching method [Liu et al., 2012b] to match the collected GPS data with the existing Google map. The start data s is calibrated to a new position, which is viewed as the estimated shooting position $\mathbf{x}_G$ from GPS.

Shooting position estimation from image. Besides the GPS data, we also leverage captured street view to estimate shooting position. Through matching captured street view with existing Google street views via three image retrieval algorithms [Jain et al., 1996, Manjunath and Ma, 1996, Pass et al., 1997], we acquire three estimated shooting positions donated as $\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{x}_3$.

When a user captures a street view, various parameters of photographing are collected from smartphone. These parameters include the direction of photographing $\overrightarrow{D}$, the angle of pitching ω_p, the angle of view ω_v and the size of photo s_p.

After a street view is captured, CrowdGIS fetches existing Google street views

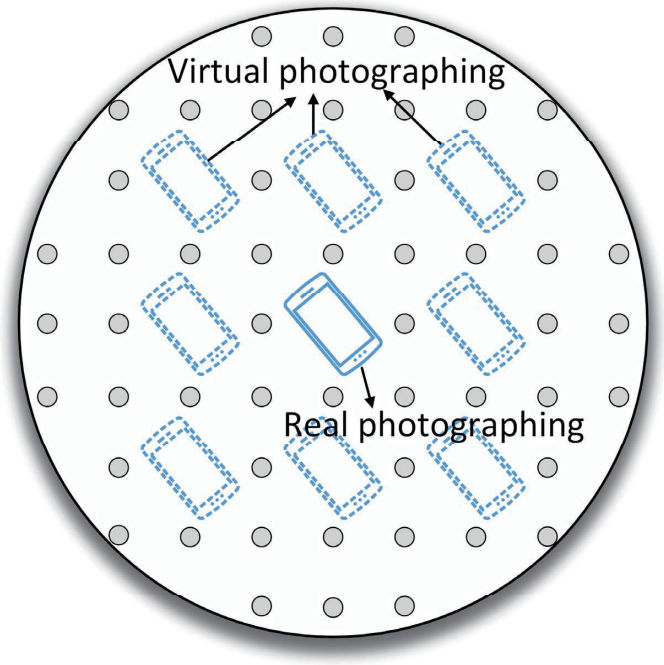


Fig. 4.3: Virtual photographing in various positions to estimate shooting position from image.

of various positions around the user with the same parameters of photographing as the user. These Google street views are downloaded through the Google Street View API. Since the system simulates photographing as the user in various positions with the same parameters, this process is called *virtual photographing*, as shown in Figure 4.3. To narrow the data size of downloaded street views, we obtain one Google street view from every position around the user with interval of 2 *meters* in a range of 50 *meters* radius.

Then CrowdGIS estimates shooting position through matching captured street view with the downloaded Google street view set. As an image can be represented by three major categories of features (i.e. colour, texture and shape), we adopt three state-of-the-art image retrieval methods [Jain et al., 1996, Manjunath and Ma, 1996,

Pass et al., 1997] to match street views. Each method outputs a most similar street view and its corresponding shooting position. Three estimated shooting positions are obtained and donated as $\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{x}_3$.

Specially, if a store is changed and different from the existing street view, our method is still effective. This is because the GPS first gives a rough position and a limited area. Then although a store is changed, the surroundings are generally unchanged (such as building, nearby stores and road signs). Street views captured by crowdsourcing will include these unchanged surroundings. The image matching method only finds the most similar street view. Thus, captured store can be matched and localized, even if it has been changed.

Joint position estimation of shooting position. To accurately localize shooting position from four estimated shooting positions, we propose a novel joint location estimation algorithm based on artificial neural network. Given several candidate shooting positions estimated from GPS and images, a naive method is directly taking the average position of all candidates as the shooting position. However, the average position may not be accurate since this method supposes a linear relationship between the candidates and real shooting position, which is impractical and problematic. Alternatively, artificial neural network (ANN) [Sutskever et al., 2014] is a superior choice to learn the unknown function between the candidates and real shooting position. Moreover, since the contributions of estimated positions are not previously known, we propose a weighted ANN to localize shooting positions.

First, we calculate the accuracy of each candidate shooting position. For the shooting position $\mathbf{x}_G$ estimated from GPS, this position is calibrated from the *start data* s with a calibration distance d. Although the distribution of GPS location

is not perfectly Gaussian because of the shape and acceleration of satellites and the atmosphere turbulence, its error can be estimated and bounded by a Gaussian distribution [Rife et al., 2004]. We have the distribution of calibration distance d:

$$f(d) = \frac{1}{\sqrt{2\pi}\sigma_G} e^{-\frac{d^2}{2\sigma_G^2}} \tag{4.3}$$

with the standard deviation denoted as σ_G. Thus, the accuracy $p_G \in [0, 1]$ of estimated shooting position $\mathbf{x}_G$ is derived from the distribution $f(d)$ of calibration distance d.

For the shooting positions ($\mathbf{x}_1, \mathbf{x}_2$ and $\mathbf{x}_3$) estimated from images, the corresponding accuracy p_1, p_2 and p_3, $p_{1,2,3} \in [0, 1]$ is defined from the Hamming Distance H [Zhang et al., 2013] of two matched street views as follows:

$$p_i = \frac{H_i}{s_p^i}, \tag{4.4}$$

where s_p^i is the size of matched street view for shooting position $\mathbf{x}_i$.

Second, we design a weighted artificial neural network to learn the relationship between four estimated shooting positions and real shooting position. In our case, since each estimated shooting position contributes to the final estimation differently, we design the input $\mathbf{I}$ of artificial neural network as weighted estimated shooting positions $\mathbf{I} = \{p_G\mathbf{x}_G, p_1\mathbf{x}_1, p_2\mathbf{x}_2, p_3\mathbf{x}_3\}$. With these weights, more accurate relationship function can be learned. The architecture of weighted artificial neural network is shown in Figure 4.4. The output of the kth neuron in the neuron layer is

$$O_k = \frac{1}{1 + exp(-S_k)}, \tag{4.5}$$

where S_k is calculated from Equation (5.15). And the objective function F of neural network is:

$$F = \frac{1}{2} \sum_{i=1}^{N_d} \sum_{s=1}^{N_L} (X_s(i) - O_s(i))^2, \tag{4.6}$$

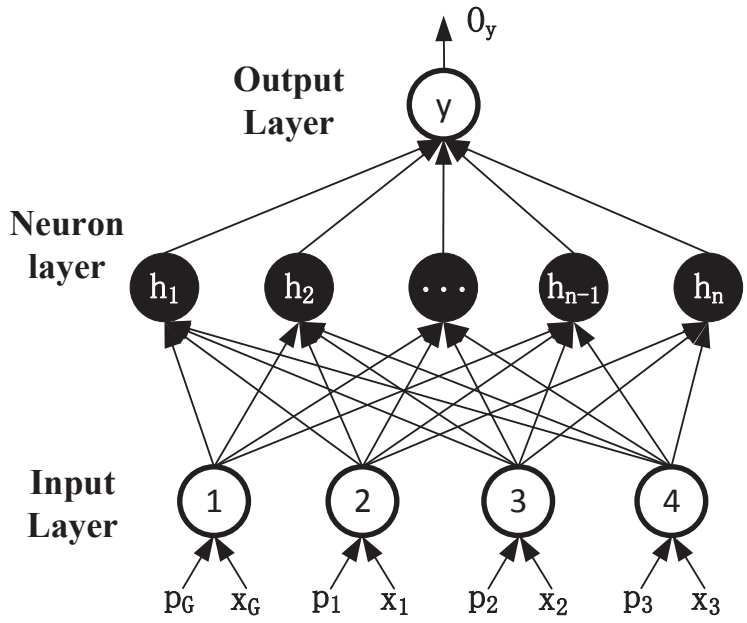
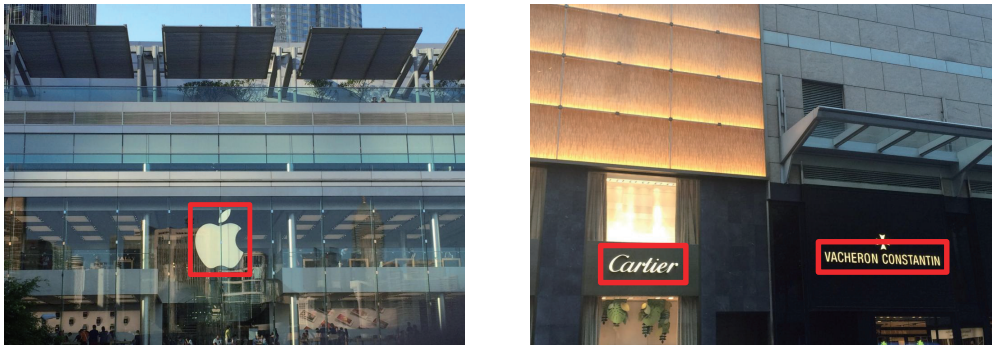


Fig. 4.4: Weighted artificial neural network architecture.

where N_d is the number of examples in the data set, N_L corresponds to the number of outputs of the neural network, X_s represents the target value corresponding to the sth neuron of the output layer. In our system, the output of weighted Neural Network is the real shooting position. After the training process, the relationship between estimated shooting positions and real shooting position can be learned. Thus, utilizing the relationship learned from the weighted ANN, CrowdGIS localizes the shooting position from estimated shooting positions.

4.4.2 Store Recognition

In this subsection, we recognize various store names from captured street views. For stores with logos, we utilize object detection technique to recognize their names. For stores without logos, we propose a text clustering method to divide and extract various store names.



(a) Recognize store name from logo. (b) Recognize store name from text.

Fig. 4.5: An example of store recognition.

Store name recognition from logo. Motivated by the observation that most stores have their unique logos, especially for chain stores, we incorporate object detection technique to recognize stores with logos. Figure 4.5 (a) gives an example. For each captured street view, we first extract the histogram of oriented gradients (HOG) and scale-invariant feature transform (SIFT) features to represent image information. Then we utilize the Locality-constrained Linear Coding (LLC) [Yuan et al., 2017a] to further encode the local features into final image histograms. The LLC utilizes the locality constraints to project each descriptor into its local-coordinate system and captures the correlations between similar local features by sharing the visual words, thus it could give better detection results compared with the traditional bag of visual words methods. For the image classification, we apply the Multiple Kernel Boosting (MKB) [Yang et al., 2011] to classify the logos. MKB is a boosted Multiple Kernel Learning (MKL) method, which combines several SVMs of different kernels [Yuan and Meng, 2017, Yuan et al., 2017b], thus it could provide better classification performance. Thus, through recognizing corresponding logos, the names of stores with logos are obtained.

Store name recognition from text. Existing text recognition methods could recognize letters, numbers and words from an image with extremely high accuracy. However, in many cases, the store name is a non-semantic word (such as "GEOX") or a combination of words (such as "bread n butter"). Moreover, the captured street view may include several stores. Hence, to accurately recognize stores without logos, we propose a text clustering technique to infer various store names based on the Manhattan World Assumption.

With the observation that most texts of store names are aligned within the three Manhattan directions, we propose a method to identify various store names from captured street views. Figure 4.5 (b) gives an example. After recognizing all letters and numbers from street view with text recognition [Yao et al., 2014] technique, we characterize each letter and number with two features: the position in street view image and color histogram. Then we cluster these letters and numbers to separate various store names with BIRCH [Zhang et al., 1996] method. Thus, letters and numbers with close distance and similar colour histogram are clustered into one group to generate a store name.

4.4.3 Store Localization

In this subsection, we illustrate the method to localize stores in the map based on the captured street views and corresponding shooting positions. We extract the position of store in the captured street view and then transfer it into the global coordinate system of the map through calculating its shooting direction.

As shown in Figure 4.6, a recognized store name is bounded with a box. We define the center position of bounding box as the position (x_{local}, y_{local}) of recognized store in the local coordinate system of captured street view. According to the principle of

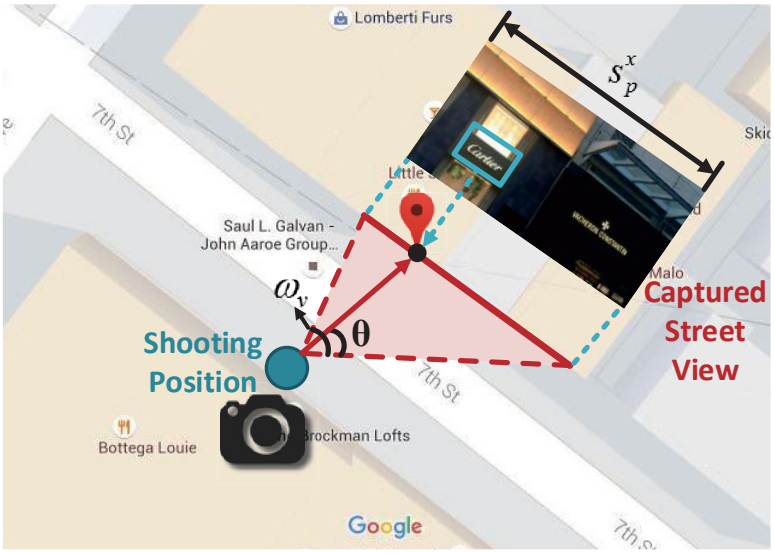


Fig. 4.6: Localizing a store in the map through calculating its deflection angle θ.

pinhole imaging, the deflection angle θ of recognized store in the global coordinate system is:

$$\theta = \omega_v - \frac{x_{local}}{s_p^x} \times \omega_v, \tag{4.7}$$

where ω_v is the angle of view and s_p^x is the length size of photo. Since the direction of photographing $\overrightarrow{D}$ collected from smartphone corresponds to half of the angle of view (i.e. $\frac{\omega_v}{2}$), the shooting direction of recognized store is obtained.

Because users capture street views within their sights, the behind concealed stores can not be captured. Thus, with the given shooting position and the angle of pitching ω_p, recognized store in the global coordinate system of the map is localized at the first intersection position of its shooting direction and walls in the Google map.

4.4.4 Map Updating

In this subsection, we update the maps with the obtained names and positions of stores from street views. Changed stores are classified into three categories, and

updated with corresponding approach. Specifically, we design a position estimation model to calculate the position of updating store based on kernel density estimate.

Defining three updating categories. Street views captured by smartphones provide reliable information about the changes of stores in the map. Based on observations in real environments, we consider three categories of updating stores: *replacing* old stores, *adding* newly-built stores and *deleting* nonexistent stores.

We classify changed stores into three updating categories by comparing their names and positions with that in the map. The process is described in Algorithm 2. Stores identified from street views are denoted as a set of tuple $\mathbf{S} = \{< n_s^i, p_s^i >\}, i = 1, 2, 3...$, where n_s^i represents the name of store i and p_s^i represents the position of store i, and stores in the existing Google map are denoted as a set of tuple $\mathbf{M} = \{< n_m^j, p_m^j >\}, j = 1, 2, 3...$, where n_m^j represents the name of store j and p_m^j represents the position of store j. For each store in both sets, we set an indicator Λ to indicate the category of updating it belongs to. We compare the name and position of each store in both sets. If the names of two stores are the same and their positions are extremely close, it proves that the store is unchanged in the map. If two store names are different in the same position, the store in the map may be replaced. For the stores identified from street views, if they cannot be matched with the map, we classify they in the newly-built store category. In contrast, if there exist unmatched stores in the map, they are viewed as potential nonexistent stores. Considering errors in the accuracy of position, we think two positions are the same if their distance is less than ε.

Updating stores in the map. To improve the robustness of our system, a significant principle for updating a store is that this store has been photographed

Algorithm 2: Classifying three updating categories

Input:
store tuple set from street view $\mathbf{S} = \{< n_s^i, p_s^i >\}$;
store tuple set from map $\mathbf{M} = \{< n_m^j, p_m^j >\}$;
Output:
category indicator vector Λ;

for *all* (S_i, M_j) **do**
    initialize Λ with *null*;
    if $n_s^i == n_m^j$ *and* $\|p_s^i - p_m^j\| \leq \varepsilon$ **then**
        $\Lambda \leftarrow$ *maintain*;
        remove the stores from tuple set **S** and **M**;
    if n_s^i *!=* n_m^j *and* $\|p_s^i - p_m^j\| \leq \varepsilon$ **then**
        $\Lambda \leftarrow$ *replace*;
        remove the stores from tuple set **S** and **M**;
for *all stores left in set* ***S*** **do**
    $\Lambda \leftarrow$ *add*;
for *all stores left in set* ***M*** **do**
    $\Lambda \leftarrow$ *delete*;

and indicated to conduct the same updating operation for sufficient times. Thus, for each category of updating stores, if a store has been indicated to conduct the same updating operation for sufficient times (more than a presupposed *support threshold*), the system conducts corresponding updating operation.

Deleting nonexistent stores. If a store in current map have not been captured in nearby street views for more than presupposed times, this store is viewed as nonexistent and deleted from current map. Because a large number of street views are collected through crowdsourcing, it is rational to assume that real existing stores can be captured in various street views.

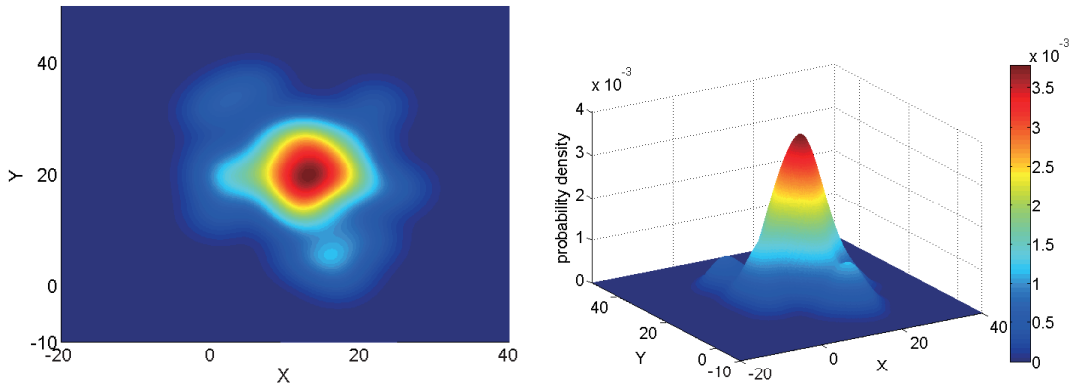
Adding newly-built stores. If a store is indicated to exist in a limited arrange for more than presupposed times, this store is viewed as newly-built and added in current map. Because indicated positions may not be exactly same in each time, we estimate its precise position based on Kernel Density Estimate (KDE) [Botev et al., 2010]. We model the distribution of the i-th indicated position of store $\mathbf{A}$ as a normal distribution. The probability density function in each position $\mathbf{x}$ is:

$$f_i(\mathbf{x}) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{\|\mathbf{x}-\mu_i\|^2}{2\sigma_i^2}}, \tag{4.8}$$

where μ_i is the position of i-th candidate of store $\mathbf{A}$, σ_i is the standard deviation of the position. Considering the probability distributions of every indicated position are mutually independent, we model the total probability density function $\hat{f}$ by using KDE:

$$\hat{f}(\mathbf{x}) = \frac{1}{nh} \sum_{i}^{n} f_i(\frac{\mathbf{x}}{h}). \tag{4.9}$$

According to Equation (4.9), we estimate the position of store $\mathbf{A}$ as the position $\mathbf{x}$ which owns the highest $\hat{f}(\mathbf{x})$. Figure 4.7 gives an example that probability density function of store position is estimated based on KDE.



(a) The 2D view of KDE. (b) The 3D view of KDE.

Fig. 4.7: Estimating probability density function of store position based on KDE.

Replacing old stores. Similarly, if a store in current map has been indicated to change into another store for more than presupposed times, this store is viewed as out-of-date. The progress of replacing a store can be considered as a combination of deleting and adding a store.

Thus, through frequently and timely updating, the map is almost always up-to-date and gracefully adapts to real environment changes.

4.5 Implementations and Evaluation

4.5.1 Experimental Methodology

We implement CrowdGIS on an Intel core i7 machine with 64GB RAM and NVIDIA TITAN X graphics card. The high-performance graphics card supports sensor data and image analysis in this work. We conduct experiments in an outdoor region covering about 3,000m×3,000m in Hong Kong. Specifically, the experiment area belongs to an urban environment, which contains 261 various stores. The stores include convenience stores, supermarkets, banks, etc.

We recruit ten volunteers to collect street views and sensing data for 1 month. Each user carries a smartphone during his daily life. The smartphones are pre-installed with a developed APP for automatically collecting sensing data while the user captures street views. When the users travel in the experiment area, they capture street views as they commonly do. The users do not need to behave intentionally for the sensing data collection. We believe that the data gathered in such way are representative for general realistic scenarios.

Besides the street views, various sensing data are automatically collected from sensors, which include GPS, accelerometer, gyroscope, and compass. The users capture one or multiple street views in one position and then continue to walk. During this process, the sensing data collection procedure is triggered for a certain period (ranging from five seconds to five minutes). The collected sensing data record the orientation and position of photographing and the subsequent moving trajectory. In our evaluation, we collect 8,471 images and 2,615 moving trajectories from volunteers.

To evaluate the location error of our system, we also collect real shooting positions of street views captured by the volunteers. For the weighted ANN utilized in shooting position localization, we build the training data set through sampling 1000 various positions in an outdoor area (different form the experiment area). And we set the number of neurons as 12 to achieve the best training performance after comparing with other neuron numbers.

OpenCV library (version 3.0) is adopted to identify stores from logos. In particular, a training data (logo images) set is built in advance to conduct the image recognition. The training images are collected from two parts, which contains 2,000 images about 100 various stores. The first part images (300 out of 2000 samples) are

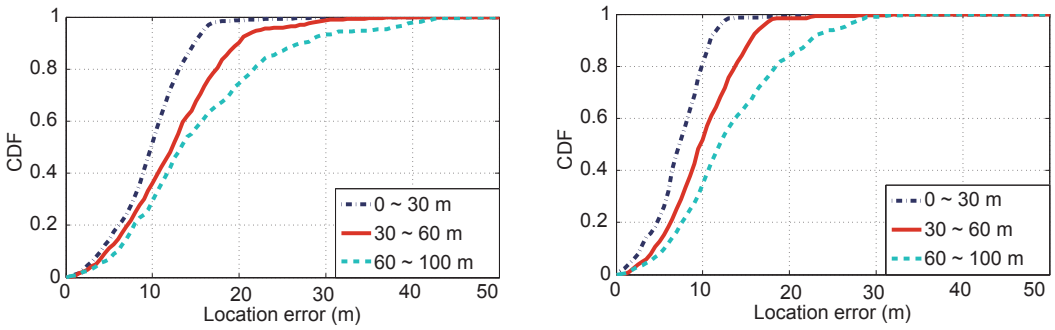
collected via photographing logos from various viewpoints in the real situation (different form the experiment area). The second part images (1700 out of 2000 samples) are downloaded from the internet to enable the generality of the images.

4.5.2 Performance Evaluation

Since CrowdGIS consists of four key modules, we evaluate each module to better understand the effectiveness and limitation of system.

Performance of street view localization. We first evaluate the localization performance of the proposed street view localization algorithm. Since street views are utilized to improve the accuracy of localization, we compare the performance of localization with and without image data. As shown in Figure 4.8 (a), localizing street views from GPS yields average accuracy of about 10.5 meters and 90th percentile accuracy of 15.7 meters when the distance between user and street view is less than 30 meters. An average accuracy of about 12.4 meters and 90th percentile accuracy of 20.2 meters are achieved when the distance is between 30 meters and 60 meters. The location accuracy degrades to 18.2 meters in average error and 27.4 meters in 90th percentile error when the distance is more than 60 meters. The results show that street view localization has less accuracy when user stands farther away from street view.

In contrast, Figure 4.8 (b) illustrates better street view localization performance when CrowdGIS combines GPS data and street views. CrowdGIS localizes street views with the average accuracy of about 7.3 meters and 90th percentile accuracy of 11.6 meters when the distance between user and street view is less than 30 meters. When the distance is between 30 meters and 60 meters, street view localization yields 9.4 meters in average error and 15.3 meters in 90th percentile error. And the location



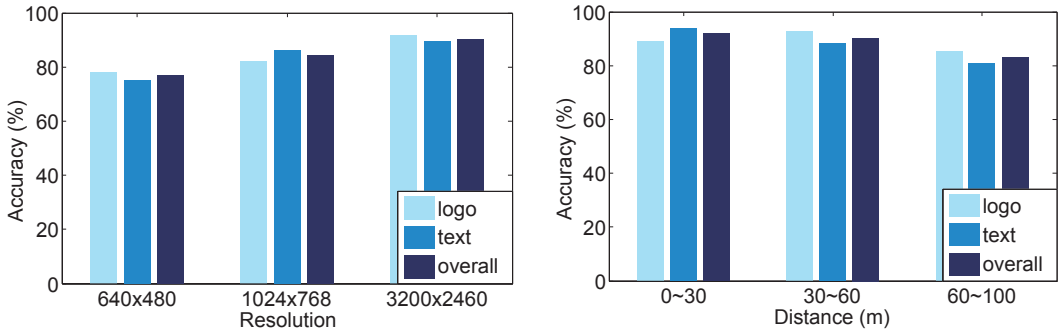
(a) Position estimation from GPS.

(b) Joint position estimation from GPS and image.

Fig. 4.8: Performance of street view localization.

accuracy degrades to 12.1 meters in average error and 22.1 meters in 90th percentile error when the distance is more than 60 meters. The high accuracy is benefitted from the stable performance of joint position estimation combining GPS and image data.

Performance of store identification. Precision of the store identification is a critical criteria of CrowdGIS. We evaluate the performance with various image resolutions and photographing distances. Figure 4.9 (a) shows the accuracy of store identification with different image resolutions. When the captured street view resolution is 640×480, store identification has an accuracy of 78.3% from logo, 75.2% from text and overall 76.9%. When the resolution is 1024×768, the accuracy is 82.1% from logo, 86.2% from text and overall 84.5%. And when the resolution is 3200×2460, CrowdGIS achieves better accuracy with 91.5% from logo, 89.4% from text and overall 90.2%. Figure 4.9 (b) shows the accuracy of store identification with different photographing distances. When the distance between user and street view is less than 30 meters, store identification attains an accuracy of 89.1% from logo, 93.8% from text and overall 91.9%. When the distance is between 30 meters and 60 meters, the accuracy is 92.8% from logo, 88.1% from text and overall 90.1%. And when the



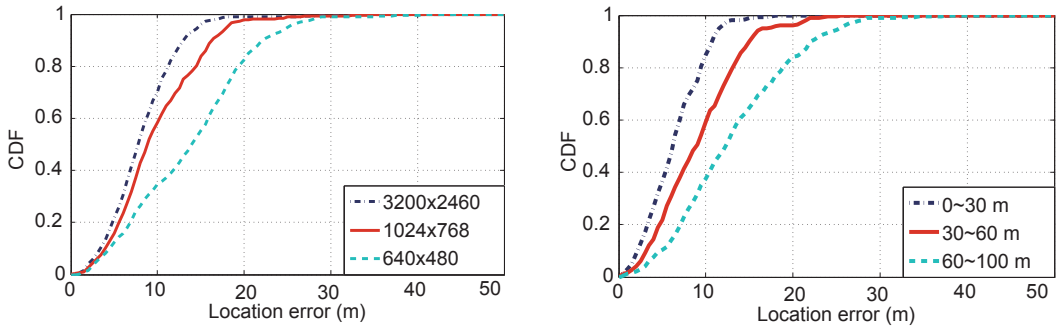
(a) Accuracy of identification vs. resolution.

(b) Accuracy of identification vs. distance.

Fig. 4.9: Performance of store identification.

distance is more than 60 meters, the accuracy degrades to 85.2% from logo, 81.1% from text and overall 82.7%. Thus, with various image resolutions and photographing distances, CrowdGIS accurately identifies various stores from captured street views.

Performance of store localization. We use location error to evaluate the performance of store localization. As shown in Figure 4.10 (a), CrowdGIS produces average accuracy of 8.4 meters and 90th percentile accuracy of 12.2 meters with resolution 3200×2460. An average accuracy of 9.1 meters and 90th percentile accuracy of 16.1 meters are achieved with resolution 1024×768. The location accuracy degrades to 14.4 meters in average error and 22.6 meters in 90th percentile error with lower resolution 640×480. Figure 4.10 (b) shows the performance with various photographing distances. Store localization yields average accuracy of 8.1 meters and 90th percentile accuracy of 11.1 meters when the distance between user and street view is less than 30 meters. An average accuracy of 8.8 meters and 90th percentile accuracy of 14.8 meters are achieved when the distance is between 30 meters and 60 meters. The location accuracy degrades to 12.5 meters in average error and 24.3 meters in 90th percentile error when the distance is more than 60 meters. Therefore in practice,



(a) Accuracy of localization vs. resolution.

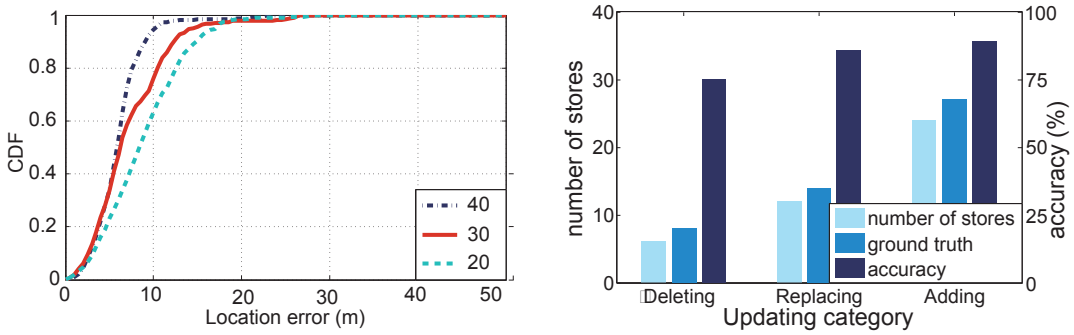
(b) Accuracy of localization vs. distance.

Fig. 4.10: Performance of store localization.

CrowdGIS localizes stores from captured street views and collected sensing data with high accuracy.

Performance of map updating. We evaluate the performance of map updating through two aspects: the accuracy of updated store position and the accuracy of updated store name. Since updating a store in the map is triggered by collecting sufficient number of candidates, we conduct experiments with different support thresholds. Figure 4.11 (a) shows the relationship between the accuracy of updated store position and support threshold. CrowdGIS achieves average accuracy of about 9.1 meters and 90th percentile accuracy of 15.2 meters when the support threshold is set to 20. If we set the support threshold to 30, the average accuracy is 6.3 meters and 90th percentile accuracy is 11.9 meters. A better performance is obtained with average accuracy of 6.1 meters and 90th percentile accuracy of 9.8 meters when the support threshold is added to 40.

Moreover, Figure 4.11 (b) shows the accuracy of updating store names in three categories. For the deleting category, 8 stores are nonexistent in the experiment area and CrowdGIS successfully detects and deletes 6 stores , which achieves 75% accuracy.



(a) Position accuracy vs. support threshold.

(b) Name accuracy vs. updating category.

Fig. 4.11: Performance of map updating.

For the replacing category, there are 14 stores are changed to other stores. CrowdGIS correctly recognizes 12 stores and replaces them with new stores, which achieves 85.7% accuracy. For the adding category, some stores are not labelled in current map and some stores are new-built, thus we find 27 stores should be added into the map. CrowdGIS succeeds in adding 24 stores into the map with 88.9% accuracy.

Figure 4.12 and Figure 4.13 give some examples of updating changed stores in the map. These experiment results demonstrate that utilizing data collected by only four volunteer users in 1 month, CrowdGIS achieves comparable accuracy and quantity of updating stores in the map. In other words, maps can be continuously updated to maintain their accuracy when abundance crowdsourcing data are collected. Thus, we envision CrowdGIS as a fundamental and indispensable reviser for existing maps to cope with store variations.

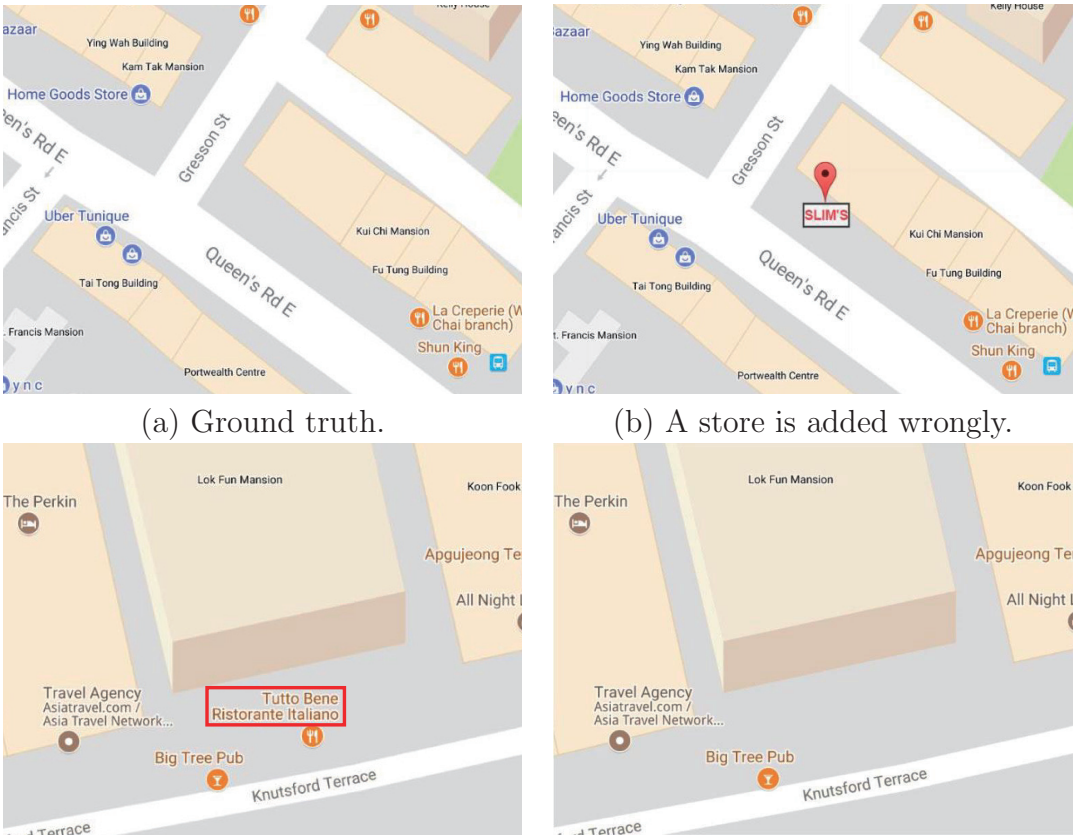
Limitation of the proposed CrowdGIS. Although the proposed CrowdGIS performs favorably against existing methods in updating digital maps with higher accuracy, it is far from perfect. It might make mistakes or give wrong information in certain cases as shown in Figure 4.14. The proposed method does not work well if a

(a) The existing map. (b) Two stores are added in the map.

Fig. 4.12: An example of adding stores.

(a) The existing map. (b) Stores are replaced and deleted.

Fig. 4.13: An example of replacing and deleting stores.



(a) Ground truth. (b) A store is added wrongly.

(c) Ground truth. (d) A store is deleted wrongly.

Fig. 4.14: Failure cases: wrongly adding and deleting stores.

store name is designed unusually [Figure 4.14 (a) and Figure 4.14 (b)], and the store name is hard to be recognized correctly. In such case, a store may be added in the map with a wrong name. In addition, when a store name is covered by an obstacle, the proposed method does not work well [Figure 4.14 (c) and Figure 4.14 (d)] and the store may be deleted in the map.

4.6 Chapter Summary

In this chapter, we propose CrowdGIS, an automatic store self-updating system for digital maps that leverages street views and sensing data crowdsourced from mobile

users. Compared with previous works, CrowdGIS represents the first attempt to automatically update stores in digital map. To realize this system, shooting positions of captured street views are first localized by a novel joint position estimation scheme. Second, store names are recognized through detecting either logo or text from street views. Third, we transfer the shooting position to the location of recognized stores in the map. Finally, CrowdGIS considers three updating categories to update them in the map based on the kernel density estimate model. We implement CrowdGIS and conduct extensive experiments in a real outdoor region for 1 month. The evaluation results demonstrate that CrowdGIS effectively accommodates store variations and maintains an up-to-date map.

Chapter 5

Urban Safety Index Inference from Location-based Data

Information about urban safety, e.g., the safety index of a position, is of great importance to protect humans and support safe walking route planning. Despite some research on urban safety analysis, the accuracy and granularity of safety index inference are both very limited. The problem of analyzing urban safety to predict safety index throughout a city has not been sufficiently studied and remains open. In this chapter, we propose U-Safety, an urban safety analysis system to infer safety index by leveraging multiple cross-domain urban location-based data. We first extract spatially-related and temporally-related features from various urban location-based data, including urban map, housing rent and density, population, positions of police stations, point of interests (POIs), crime event records, and taxi GPS trajectories. Then, these features are fed into a novel sparse auto-encoder (SAE) framework with feature correlation constraint to obtain the final discriminative feature representation. Finally, we design a new co-training-based learning method, which consists of two separated classifiers, to calculate safety index accurately. We implement U-Safety and conduct extensive experiments by utilizing various real data sources obtained in

New York City. The evaluation results demonstrate the advantages of U-Safety over other methods.

5.1 Overview

Urban safety information of an area, such as the occurrence of crime and emergency, is of great importance to support urban safety control and protect humans from danger. A traveller urgently needs such valuable information to book a hotel before his arrival, and to choose a safe walking route in a time slot when he stays in a new and unfamiliar city. Although safety is the first-priority factor in guiding our daily lives, people often have very limited information about how safe/dangerous of an area in a city, especially for travellers. Some cities, e.g. New York City, only provide very rough safety heat map by pinning occurred crimes in a city map [cri]. Such safety heat map cannot unveil the impact of distinct factors (e.g., temporally-related or spatially-related) and thus cannot present accurate safety level of an area nor predict it. Therefore, in a smart city, it is crucial to conduct urban safety analysis to build a periodically updated safety-index map in fine granularity for travellers and citizens.

Despite a few works have been devoted to urban safety analysis, accurate assessment and prediction of safety index of city areas have not been realized yet. Previous work can be classified into two types. The first type can predict the safety of an area through image analysis. Ordonez et al. [Ordonez and Berg, 2014] apply computer vision techniques to predict perceptual characteristics of urban environments in terms of wealth, uniqueness, and safety from a sampled data set of street view

images. Arietta et al. [Arietta et al., 2014] present a method for predicting relationships between visual elements and city attributes such as crime rates, theft rates and danger perception from street-level images of a city. Khosla et al. [Khosla et al., 2014] propose an approach to look beyond the immediately visible urban scene using visual elements for predicting potential crime rate for an area. The other type is based on classical mathematical models. These methods [Christin et al., 2013, Matei et al., 2001, Traunmueller et al., 2016] develop various models to describe safety index from factors like street geometry, traffic flow and human behavior, based on a number of empirical assumptions and parameters.

While existing studies have tried to explore the possibility of urban safety analysis, accurate assessment and prediction of safety index of each position in urban area deserves more attention. First, although street images can be leveraged to predict crime rate or safety of an area, these methods [Arietta et al., 2014, Khosla et al., 2014, Ordonez and Berg, 2014] cannot achieve high fine granularity or high prediction accuracy. Street images in some places may be nonexistent and the prediction of urban safety is not dynamic or low-frequency. Because various categories of urban location-based data can be collected nowadays, it is promising to analyze the relationships between multiple factors and urban safety to accurately and dynamically predict the safety index of a position. Second, classical mathematical models utilized in methods [Christin et al., 2013, Matei et al., 2001, Traunmueller et al., 2016] need some empirical assumptions and parameters. But these assumptions and parameters might not be applicable to all urban environments, which limit extensive applications of these methods. As a consequence, we ask the following question: *Is it feasible to analyze the relationship between multiple factors and urban safety to dynamically*

predict the safety index in smart city with high fine granularity?

In this chapter, we provide an affirmative answer through the systematic design and implementation of *U-Safety*, which enables fine-grained safety index throughout a city to be accurately analyzed and inferred from various categories of urban location-based data. Different from current schemes, the proposed U-Safety utilizes the state-of-the-art machine learning and data mining techniques to learn the relationship between safety index and various features extracted from multiple heterogeneous data sources. Multiple historical cross-domain urban location-based data are analyzed, including urban map, taxi trip data, housing rent and density data, population, positions of police stations, crime events, and POIs (Point Of Interests). U-Safety can comprehensively and objectively reflect city dynamics from a data perspective. To the best of our knowledge, our work represents the first attempt to cope with urban safety dynamics to build a periodically updated safety index map from multiple cross-domain urban location-based data.

Implementing such a functional system entails distinct challenges. The first challenge is to identify discriminative features from multiple cross-domain data sources. The second one is how to fuse heterogeneous features across various modalities. Equally treating these features or simply concatenating them cannot obtain a high accuracy for many data mining tasks [Bengio et al., 2013, Ngiam et al., 2011, Srivastava and Salakhutdinov, 2012, Yuan et al., 2015]. Third, it is not trivial to propose an accurate model to analyze and predict safety index, which is influenced by multiple factors with a nonlinear relationship.

To address the above challenges, we make the following contributions in this chapter:

- We propose the *U-Safety* system that can analyze multiple cross-domain urban location-based data to infer safety index throughout a city with high granularity.
- We conduct effective *feature extraction*, e.g. spatially-related and temporally-related features, which contributes to not only our system but also the extensive applications of urban dynamics analysis.
- We present a novel *feature fusion* method to feed fused features into a spatial or temporal classifier, instead of treating features equally, leading to a high classification and inference accuracy.
- We design a *co-training-based learning* approach to inferring safety index of a position. The approach consists of two classifiers respectively modelling the spatial and temporal features which influence the safety index.
- In addition, we evaluate our system using a large amount of multiple historical cross-domain urban location-based data collected in New York City. The results illustrate that U-Safety effectively accommodates urban safety dynamics and infers accurate safety index.

The remainder of this chapter is organized as follows. We first describe an overview of U-Safety system in Section 5.2. Then, we detail each module of our system in Section 5.3. The implementations and evaluation are illustrated in Section 5.4. Finally, we conclude this chapter in Section 5.5.

5.2 System Design

U-Safety system analyzes multiple cross-domain urban data to infer safety index throughout a city with high fine granularity. The potential relationship between

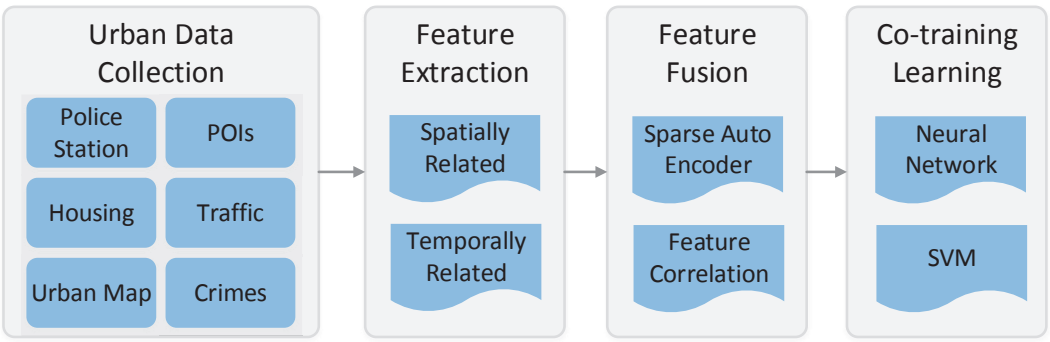


Fig. 5.1: The architecture of U-Safety system.

multiple factors and urban safety is investigated. Figure 5.1 sketches the U-Safety system architecture. In the following, we describe four modules of the system briefly.

Urban Data Collection. We collect a large amount of multiple historical cross-domain urban data (more than 300G) from authoritative official organizations in New York City. These data record various aspects of city dynamics that influence the safety index, including urban map, housing rent and density data, population, positions of police stations, POIs, crime event records, and GPS trajectories generated by over 13,000 taxis.

Feature Extraction. We identify diverse features from collected data and divide them into two categories, i.e. spatially-related features (such as POIs, distance to a police station, and housing rent) and temporally-related features (such as traffic flow, and human mobility). Various effective features are utilized in corresponding classifier in the co-training learning framework to achieve a good classification and inference performance.

Feature Fusion. After extracting various features, we design a novel feature fusion technique to generate more effective features. The extracted features are fused

through leveraging Sparse Auto-Encoder (SAE) [Shin et al., 2013] to learn a representation for features. By considering feature correlation constraint, a new cost function is designed to decrease the correlation between various features based on canonical correlation analysis (CCA) [Sun et al., 2011].

Co-training Learning. We present a co-training-based learning approach to improve the inference accuracy, which includes two separated classifiers modeling the spatial and temporal features respectively which influence the safety index. One is a spatial classifier based on an artificial neural network (ANN) [Sutskever et al., 2014], involving spatially-related features to model the spatial relationship between crime events in various locations. The other is a temporal classifier based on a support vector machine (SVM) [Huang et al., 2012], which takes temporally-related features as input to model the temporal dependency of crime events for a location.

5.3 Safety Index Map Construction System

In this section, we illustrate the proposed method to analyze multiple cross-domain urban data and predict the safety index throughout a city.

5.3.1 Urban Data Collection

In this work, we collect a large amount of cross-domain urban data in New York City. These data include urban map, housing rent and density data, population, positions of police stations, POIs, crime event records, and GPS trajectories. We describe part of data as follows.

Safety Index. Safety index (SI) is a number used to communicate to the public how safe an area is currently. As the SI increases, crimes will occur with a decreasing

Table 5.1: SI values, descriptors, and color codes

SI	**Values Levels of Safety Concern**	**Colors**
$0-20$	Dangerous (D)	Maroon
$21-40$	Unsafe (U)	Red
$41-60$	Risky (R)	Orange
$61-80$	Moderate (M)	Yellow
$81-100$	Safe (S)	Green

possibility. SI values are divided into ranges, and each range is assigned a descriptor and a color code. Computing the SI requires crime statistics from official department. In this chapter, crime statistics data are collected from FBI and the U.S. justice department, including willful homicide, rape, robbery, aggravated assault, larceny, arson, and traffic accident. We calculate the safety index as follows:

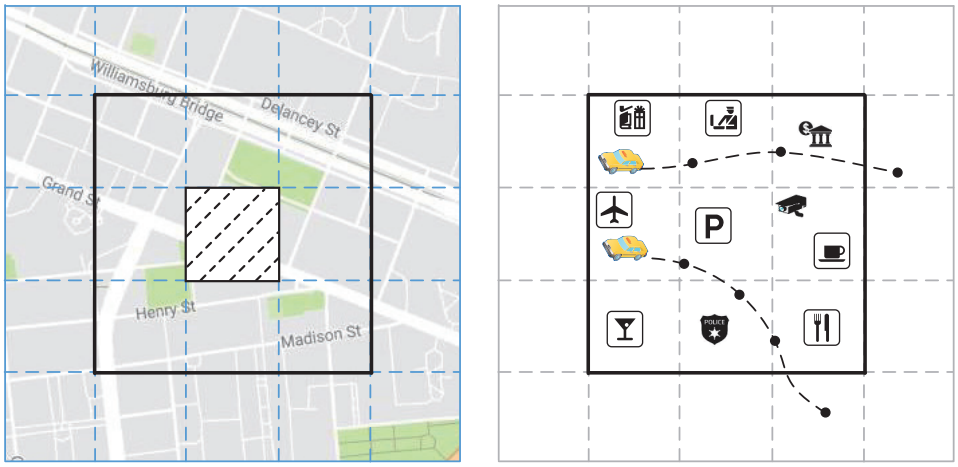
$$SI = (1 - R_C) * 100, \tag{5.1}$$

where R_C is crime rate per 100,000 inhabitants and has been normalized. As shown in Table 5.1, the descriptor of each SI level is viewed as the class to be inferred, i.e., $\mathcal{C} = \{D, U, R, M, S\}$, and the color is adopted in the following visualization figures.

Trajectory. A spatial trajectory is a sequence of time-ordered spatial points: $p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n$, where each point has a geospatial coordinate set loc and a timestamp t, $p = (loc, t)$. In this work, we collect GPS trajectories generated by over 13,000 taxis in New York City.

POI. A point of interest (POI) is a specific location (such as a cinema and school) in physical world, which has a name, coordinate, and category. Information about POIs is gathered from digital urban map.

Grid and Affecting Region. To analyze the safety index throughout a city,



(a) Grid g and affecting region R. (b) Data of POI and trajectory.

Fig. 5.2: Illustration of the grid g, affecting region R, POI, and trajectory.

we divide the city into disjointed grids (e.g., the granularity of $200m \times 200m$), as illustrated in Figure 5.2 (a). Each grid g has a geospatial coordinate loc and a SI label c to be inferred. We believe the safety index of a grid would be influenced by the urban data collected in the affecting region R that consists of the grid and its eight neighbour grids, as illustrated in Figure 5.2 (b).

5.3.2 Feature Extraction

In this subsection, we extract various effective features from collected urban data. These features are divided into two categories: spatially-related and temporally-related features.

Police-station-related Features F_s. The number of police stations and the distance to them in a region have a strong correlation with security situation in this area, thus contributing to the safety index inference of the region. Police-station-related features belong to spatially-related features. Accordingly, we identify the following two features for each grid: (1) the number of police stations in affecting

Table 5.2: Various categories of POIs

C_1: Transportation spot	C_7: Park
C_2: Hospital	C_8: Education
C_3: Factory	C_9: Entertainment
C_4: Shopping mall and supermarket	C_{10}: Company
C_5: Food and beverage	C_{11}: Hotel
C_6: Sport	C_{12}: Residential area

region R: f_n, and (2) the distance to the nearest police station for each grid g: f_d.

POI-related Features F_p. The category of POIs and their density in a region describe the function and social environment of this region, therefore providing a good complement to analyze urban safety. POI-related features are spatially-related features. Thus, we separate POIs into various categories based on the digital urban map database, and identify the number of POIs f_p in each category as features for each grid. Various categories we studied in this work are shown in Table 5.2.

Housing-related Features F_r. Housing reflects the economic condition and population density of a region, which has a potential effect on the crime rate. Housing-related features belong to spatially-related features. Therefore, we identify the following two features for each grid: (1) the number of housing in affecting region f_h, and (2) the average rent of housing in affecting region f_r. These features are extracted from housing information published by the department of housing preservation and development.

Traffic-related Features F_t. It is widely believed that traffic flow has an influence on urban safety [Tian et al., 2013]. Traffic-related features are related to temporal features. Thus, we identify the following two features for each grid. These features are generated from the spatial trajectories of vehicles traversing the grid in

the past hour: (1) the number of vehicles in affecting region f_t, and (2) the average speed of vehicles f_v. For a spatial trajectory generated by a vehicle, we extract locations which fall in the affecting region (i.e., $p \in R$). We calculate the speed of each vehicle through the distance between each two sequential points and the corresponding timestamps. Then the average speed of all vehicles f_v in R is computed as follows:

$$f_v = \frac{1}{f_t} \sum_{f_t} \frac{\sum_{p_i \in R} Dist(p_i, p_{i+1})}{\sum_{p_i \in R} |p_{i+1}.t - p_i.t|}. \tag{5.2}$$

Human-mobility-related Features F_h. Human mobility implies valuable information about traffic flow, and function and social environment of a region, contributing to urban safety index inference. Human-mobility-related features are temporally-related features. Therefore, we identify the following two features for each grid: (1) the number of people arriving at affecting region f_a, and (2) the number of people leaving from affecting region f_l. These features are extracted from the taxi trajectories which record the pickup and drop off locations in each trip.

5.3.3 Feature Fusion

In this subsection, we illustrate the feature fusion method to generate more effective feature representation of features extracted in the feature extraction step. To learn maximally correlated feature representation, we design a novel cost function in the SAE framework with feature correlation constraint.

Sparse Auto-Encoder (SAE). SAE [Shin et al., 2013] is a symmetrical and unsupervised neural network to automatically learn effective feature representation. It is realized by minimizing the reconstruction error between the input data at the encoding layer and its reconstruction at the decoding layer. The structure of SAE is

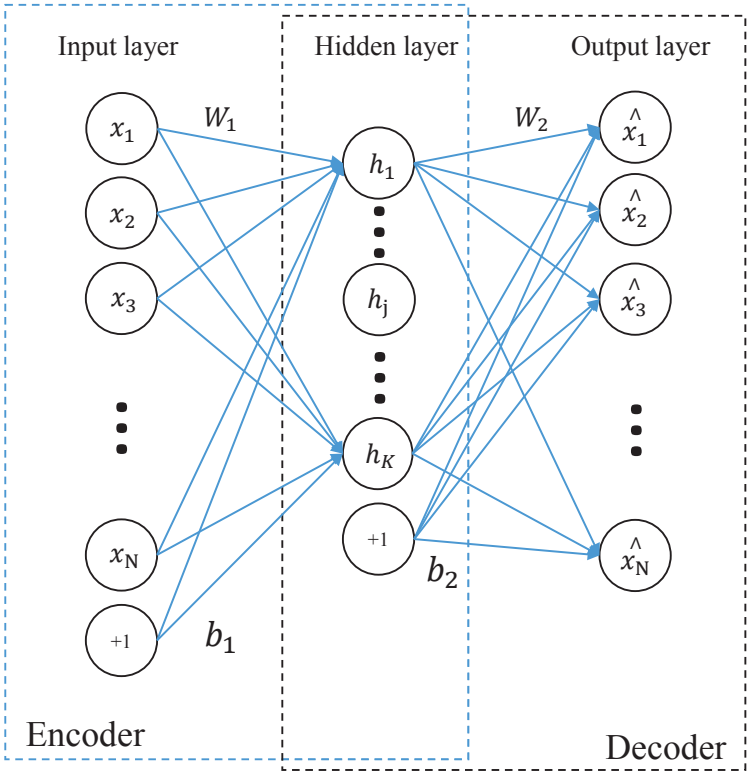


Fig. 5.3: The structure of SAE model.

illustrated in Figure 5.3.

During the encoding step, an input vector $x_i \in \mathbb{R}^M (i = 1, ..., N)$ denotes a feature extracted in the feature extraction step. The input vector x_i is processed by applying a linear mapping and a nonlinear activation function to the network,

$$h_j = f(x) = g(W_1 x_i + b_1), \tag{5.3}$$

where $W_1 \in \mathbb{R}^{K \times M}$ is a weight matrix, $b_1 \in \mathbb{R}^K$ denotes the encoding bias. M and N are the dimension and numbers of features, respectively. $g(x)$ denotes the logistic sigmoid function $(1 + exp(-x))^{-1}$.

In the decoding step, we decode a hidden representation h_j using another linear

decoding matrix $W_2 \in \mathbb{R}^{K \times M}$ as follows:

$$\hat{x}_i = f(x) = g(W_2^T h_j + b_2), \tag{5.4}$$

where $b_2 \in \mathbb{R}^M$ is the decoding bias and $\hat{x}_i$ represents the reconstructed feature of x_i.

The objective of SAE is to learn parameters W_1, W_2, b_1, b_2 to generate the hidden layer $\{h_1, ..., h_K\}$, which is a new feature representation of the input original features. This objective is realized by minimizing a cost function as follows:

$$\begin{aligned} J_{SAE} = & \frac{1}{2} \sum_{i=1}^{N} \|x_i - \hat{x}_i\|^2 + \frac{\alpha}{2}(\|W_1\|^2 + \|W_2\|^2) \\ & + \beta \sum_{j=1}^{K} KL(\rho \| \hat{\rho}_j), \end{aligned} \tag{5.5}$$

where the parameter α controls the penalty term facilitating weight decay, and β represents the sparsity penalty control parameter.

The first term in Equation (5.5) is an average sum-of-squares error term which describes the differences between input x_i and reconstruction $\hat{x}_i$. The second term is a weight decay term that tends to decrease the magnitude of the weight, and helps to prevent overfitting. The third term indicates sparsity constraint term, which utilizes the Kullback-Leibler divergence [Van Erven and Harremos, 2014] to provide the sparsity connection constraint between layers in SAE. In this term, $KL(\rho \| \hat{\rho}_j)$ denotes the K-L divergence between ρ and $\hat{\rho}_j$ with the definition as follows:

$$KL(\rho \| \hat{\rho}_j) = \rho log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) log \frac{1 - \rho}{1 - \hat{\rho}_j}, \tag{5.6}$$

where the parameter ρ represents the sparsity parameter chosen to be a small positive value near zero to impose the sparsity constraint to the SAE model, and the parameter $\hat{\rho}_j$ is the average activation of jth input vector h_j over the N training data. This

penalty function has the property that $KL(\rho\|\hat{\rho}_j) = 0$ if $\rho = \hat{\rho}_j$, and otherwise, it increases monotonically as $\hat{\rho}_j$ diverges from ρ.

Typically, back-propagation algorithm [Phansalkar and Sastry, 1994] is used to solve Equation (5.5). Thus we can obtain feature representation h for input feature x with the SAE model.

Feature Correlation Constraint. The traditional SAE model treats each input feature individually and independently without considering any correlation between various categories of features. The feature representation learned by this model may not accurately describe input original features, which will decrease the performance of learning process. Therefore, leveraging intrinsic correlations concealed in input features can effectively help discover more precise feature representation.

It is reported that canonical correlation analysis (CCA) [Hotelling, 1936] can successfully find the correlations between two sets of multi-dimensional variables in various applications [Andrew et al., 2013, Hardoon et al., 2004, Sun et al., 2011]. In the U-Safety, the safety index of a position at $t+1$ time slot is inferred from various data collected during previous t time slots. Thus, we define each input feature (vector) F_i as a set of variables by extending to time series: $F_i = [F_i^1, F_i^2, ..., F_i^t]^T$, where t denotes the time slot. Without loss of generality, let $F_1 \in \mathbb{R}^{n_1}$ and $F_2 \in \mathbb{R}^{n_2}$ denote two input vectors with covariances $\sum_{11}$ and $\sum_{22}$, and cross-covariance $\sum_{12}$. CCA finds two projection vectors ω_1 and ω_2, such that the correlation coefficient ϕ between $F_1$and F_2 is maximized:

$$\begin{aligned} \phi &= \max_{\omega_1,\omega_2} corr(\omega_1' F_1, \omega_2' F_2) \\ &= \max_{\omega_1,\omega_2} \frac{\omega_1' \sum_{12} \omega_2}{\sqrt{\omega_1' \sum_{11} \omega_1 \omega_2' \sum_{22} \omega_2}}. \end{aligned} \tag{5.7}$$

Since ϕ is invariant to the scaling of ω_1 and ω_2, this optimization problem can be

formulated equivalently as:

$$\begin{aligned} \max_{\omega_1,\omega_2} \quad & \omega_1' \sum\nolimits_{12} \omega_2 \\ \text{subject to} \quad & \omega_1' \sum\nolimits_{11} \omega_1 = 1, \\ & \omega_2' \sum\nolimits_{22} \omega_2 = 1. \end{aligned} \tag{5.8}$$

To obtain the maximum of correlation, multiple pairs of vectors (ω_1^i, ω_2^i) can be found with an orthonormality constraint that subsequent projections are uncorrelated with previous ones, i.e. $\omega_1^i \sum_{11} \omega_1^j = \omega_2^i \sum_{22} \omega_2^j = 0$ for $i < j$. Assembling the top $k \leq min(n_1, n_2)$ projection vectors ω_1^i into the columns of a matrix $A_1 \in \mathbb{R}^{n_1 \times k}$, and combining similarly ω_2^i into $A_2 \in \mathbb{R}^{n_2 \times k}$, we can identify the top k projections as follows:

$$\begin{aligned} \max_{A_1,A_2} \quad & tr(A_1' \sum\nolimits_{12} A_2) \\ \text{subject to} \quad & A_1' \sum\nolimits_{11} A_1 = I, \\ & A_2' \sum\nolimits_{22} A_2 = I, \end{aligned} \tag{5.9}$$

where $tr(\,\cdot\,)$ finds the trace of matrix. Then the optimal objective value is derived by the sum of the top k singular values of T:

$$T \triangleq \sum\nolimits_{11}^{-1/2} \sum\nolimits_{12} \sum\nolimits_{22}^{-1/2}. \tag{5.10}$$

After obtaining the optimal correlation coefficient of each two input features, we incorporate the feature correlation into the feature fusion by minimizing

$$-\gamma \sum_i \sum_{j>i} FC(x_i, x_j) = -\gamma \sum_i \sum_{j>i} tr(A_i' \sum\nolimits_{ij} A_j), \tag{5.11}$$

where γ is the correlation penalty control parameter.

SAE with Feature Correlation Constraint. To obtain discriminative SAE, the feature correlation constraint is introduced into SAE to serve as a discriminative

term. Thus, in order to emphasize the correlation between features, we modify the traditional SAE by adding the feature correlation constraint and design a novel cost function as follows:

$$\begin{aligned} J_{SAEFC} = & \frac{1}{2}\sum_{i=1}^{N}\|x_i - \hat{x}_i\|^2 + \frac{\alpha}{2}(\|W_1\|^2 + \|W_2\|^2) \\ & + \beta\sum_{j=1}^{K} KL(\rho\|\hat{\rho}_j) - \gamma\sum_{i}\sum_{j>i} FC(\hat{F}_i, \hat{F}_j). \end{aligned} \tag{5.12}$$

The first three terms in Equation (5.12) share the same definitions as in Equation (5.5). The last term presents the correlation between each two features, where γ controls the contribution of feature correlation constraints. $\hat{F}_i$ represents the feature in output layer that is a set of variables by extending to time series. Our proposed novel feature fusion method emphasizes the feature correlation, which can generate effective feature representation for features extracted from different domains with multiple modalities. In the learned feature representation, features from different domains are maximally correlated. This is because each instance (i.e., grid g) has a unique class label (i.e., safety index c). Various features for each instance should be correlate to the same label. Therefore, the correlation between various features is significantly strengthened by analyzing potential relation pattern.

We calculate the gradient of Equation (5.12) with the back-propagation algorithm [Phansalkar and Sastry, 1994] to update the parameters W_1, W_2, b_1, b_2 as follows:

$$\begin{aligned} W_1 &= W_1 - \mu\frac{\partial J_{SAEFC}}{\partial W_1}, \quad b_1 = b_1 - \mu\frac{\partial J_{SAEFC}}{\partial b_1}, \\ W_2 &= W_2 - \mu\frac{\partial J_{SAEFC}}{\partial W_2}, \quad b_2 = b_2 - \mu\frac{\partial J_{SAEFC}}{\partial b_2}, \end{aligned} \tag{5.13}$$

where μ is the learning rate.

5.3.4 Co-training Learning

In this subsection, we infer the safety index throughout a city through a co-training learning method. This method integrates two classifiers respectively modeling the spatial and temporal features which both influence the safety index.

Co-training: Co-training is a semi-supervised learning technique, where we have access to labeled as well as unlabeled data [Blum and Mitchell, 1998]. It assumes that each instance can be divided into two distinct views which provide different and complementary information about this instance. Ideally, features of each instance in two views are conditionally independent given the class label. Particularly, the co-training approach can achieve a better inference performance when one classifier successfully labels data and the other one misclassifies data [Nigam and Ghani, 2000].

In our U-Safety system, data in the training set are all labeled, which belongs to the supervised learning. Because multi-view semi-supervised learning can be viewed as a general problem of multi-view supervised learning, the idea of co-training is also adapted in our system. Consequently, after generating effective feature presentation, we present a learning approach utilizing the co-training framework to improve the inference accuracy of safety index, which includes two separated classifiers modeling the spatial and temporal factors respectively.

In the co-training framework, one classifier is a *Spatial Classifier* (SC), which takes spatially-related features as input to model the spatial relationship between crime events in various locations. The other classifier is a *Temporal Classifier* (TC), involving temporally-related features to model the temporal dependency of crime events for a location.

We first train these two classifiers with corresponding sets of features separately.

In the inference stage, we determine the safety index of a position based on the product of two probability scores (P_{SC} and P_{TC}) generated by the two classifiers, as follows:

$$c = \arg\max_{c_i \in \mathcal{C}} P_{SC}^{c_i} \times P_{TC}^{c_i}, \tag{5.14}$$

where c is the inferred safety index of a position and c_i is the ith safety index level in the level set $\mathcal{C}$.

Spatial Classifier (SC): The spatial classifier infers the safety index c of a grid utilizing corresponding spatially-related feature representation. These features include F_s, F_p, and F_r, donating the police-station-related features, POI-related features, and housing-related features.

A shallow neural network with three layers trained with Back-Propagation (BP) has been proven that it can approximate any nonlinear function with arbitrary precision [Cybenko, 1992]. Compared with linear models, the artificial neural network (ANN) [Sutskever et al., 2014] has advantages on nonlinear fitting and is more suitable to learn the complicated relationship in the model [Dreiseitl and Ohno-Machado, 2002, Gevrey et al., 2003, Khosravi et al., 2011]. Therefore, because of these advantages, we apply the ANN to learn the underlying relationship between various features and the safety index.

ANN is a computational multi-layer model based on the structure and functions of interconnected neurons. The expression of the weighted sum to the k-th neuron in the j-th layer ($j \geq 2$) is given by

$$S_{j,k} = \sum_{i=1}^{N_{j-1}} (\omega_{j-1,i,k} I_{j-1,i}) + b_{j,k}, \tag{5.15}$$

where $I_{j-1,i}$ is the information from the i-th neuron in the $(j-1)$-th layer, $b_{j,k}$ is the bias term and N_{j-1} is the number of neurons in the $(j-1)$-th layer. The output of

the kth neuron in the j-th layer ($j \geq 2$) is

$$O_{j,k} = \frac{1}{1 + exp(-S_{j,k})}. \tag{5.16}$$

And the objective function F of ANN is:

$$F = \frac{1}{2} \sum_{i=1}^{N_d} \sum_{s=1}^{N_L} (X_s(i) - O_s(i))^2, \tag{5.17}$$

where N_d is the number of examples in the data set, N_L corresponds to the number of outputs of ANN, X_s represents the target value corresponding to the sth neuron of the output layer.

The inputs of ANN are spatially-related feature representation F_s, F_p, and F_r in the affecting region (i.e., a grid g and its eight neighbour grids). The output of ANN is the safety index level c of grid g. After the training process, the relationship between spatially-related features and the safety index will be learned. In the inference process, given spatially-related feature representation in a affecting region, the safety index level c of grid g and its corresponding probability score P_{SC} can be obtained.

Temporal Classifier (TC): The temporal classifier infers the safety index c of a grid utilizing corresponding temporally-related feature representation. These features include F_t, and F_h, donating the traffic-related features, and human-mobility-related features.

The support vector machine (SVM) [Cao and Tay, 2003] can provide strong robustness by choosing an appropriate generalization grade [Cauwenberghs and Poggio, 2000]. The kernel utilized in SVM implicitly contains a non-linear transformation, which makes data linearly separable in a new feature space [Huang et al., 2012]. Therefore, based on these advantages, we apply the SVM to learn the underlying relationship between various features and safety index.

SVM is a supervised machine learning method on the foundation of statistical learning. The basic idea of SVM is to find the optimal hyper-plane that maximizes the margin in the feature space to separate the points of diverse classes. One way to pose this optimization task is as follows:

$$\begin{aligned} \min_{\vec{\omega},b} \quad & \frac{\|\vec{\omega}\|^2}{2} \\ \text{subject to} \quad & y_i(\vec{\omega} \cdot \vec{x_i} - b) \geq 1, \forall i, \end{aligned} \tag{5.18}$$

where $\vec{\omega}$ is the normal vector to the hyper-plane, x_i is the ith input training example, and y_i is the corresponding output of the SVM for the ith training example.

To achieve high accuracy, we choose the RBF kernel as the kernel function of SVM:

$$K(x, x') = exp(-\frac{\|x - x'\|^2}{2\sigma^2}), \tag{5.19}$$

where x and x' represent feature vector.

The inputs of SVM are temporal-related feature representation F_t, and F_h in the affecting region (i.e., a grid g and its eight neighbour grids). The output of SVM is the safety index level c of grid g. Since we have five different classes, we adopt a multi-class SVM technique. In our system, the multi-class problem is reduced to multiple binary classification problem. We build five binary classifiers which distinguish one certain type with the rest. In the inference process, for each instance given temporal-related feature representation in a affecting region, we apply the two-class SVM to obtain the safety index level c of grid g and its corresponding probability score P_{TC}. Especially, the probability score P_{TC} of the inferred safety index level c of grid g is derived by mapping the SVM output into probability through an additional parameter-trained sigmoid function [Lin et al., 2007, Platt et al., 1999].

5.4 Implementations and Evaluation

In this section, we present the implementation and evaluation of our proposed U-Safety system.

5.4.1 Experimental Methodology

We carry out experiments of safety index assessment and prediction on an Intel core i7 machine with 32GB RAM and NVIDIA TITAN X graphics card. A large amount of multiple historical cross-domain urban data (more than 300G) are collected from authoritative official organizations in New York City. These data include urban map, housing rent and density data, population, positions of police stations, POIs, crime event records, and GPS trajectories generated by over 13,000 taxis. These taxis are equipped with GPS which can be viewed as a large number of mobile sensors measuring the travelling speed on the road. The trajectory data can also provide the pick-up and drop-off locations in each trip. The total distance of all trips reaches 216 million kilometers, and the amount of accessing points is over 78 million. Because taxis contribute about 30 percent of traffic flow in New York City [Salon, 2009], the data is big enough to represent the traffic patterns there. In this chapter, about two-thirds of data are used to train our models, and the other part are used as the ground truth to compare the performance of our methods. The whole city is divided into disjointed grids with the granularity of $200m \times 200m$. Each grid has a unique safety index label to be inferred.

Table 5.3: Performance of feature extraction

Features	Precision	Recall
F_s	0.351	0.372
F_p	0.433	0.418
F_r	0.318	0.343
$F_s + F_t$	0.559	0.532
$F_s + F_p$	0.592	0.585
$F_s + F_p + F_t + F_h$	0.709	0.726
$F_s + F_p + F_r + F_t + F_h$	0.781	0.770

5.4.2 Performance Evaluation

Performance of Feature Extraction. We first evaluate the effectiveness of extracted features by comparing the performance of individual features and their various combinations. Table 5.3 shows the precision and recall of safety index inference with different feature sets. If all five categories of extracted features are utilized, the precision and recall of safety index inference can reach 78.1% and 77.2% respectively. In addition, when we add one more feature set into the model, the system will achieve a significant improvement on both precision and recall. This is because all features we extracted are effective and relevant to the safety index. With more features introduced into the system, better inference performance can be obtained.

Performance of Feature Fusion. To further study the performance of our approach, we evaluate the performance of feature fusion. We compare the precision and recall of safety index inference with or without feature fusion. As shown in Figure 5.4, the inference of safety index throughout the city can achieve a better performance when we apply the feature fusion module before the co-training stage. The precision and recall of safety index inference can reach 78.1% and 77.2% respectively with

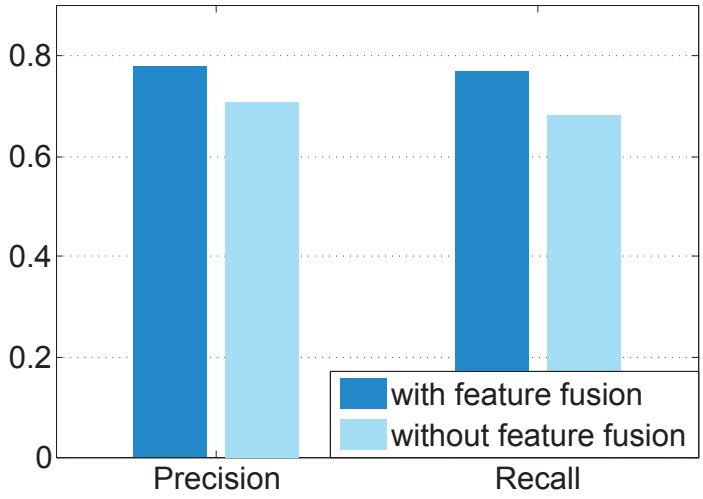


Fig. 5.4: Performance of feature fusion.

the feature fusion. And the precision and recall of safety index inference decrease to 70.9% and 68.2% respectively without the feature fusion. This is because each instance (i.e., grid g) has a unique class label (i.e., safety index c). For each instance, various features extracted from different domains with multiple modalities should be correlate to the same label. Our proposed novel feature fusion method emphasizes the feature correlation, which can generate effective feature representation for features. In the learned feature representation, features from different domains are maximally correlated. The correlation between various features is significantly strengthened by analyzing potential relation pattern. Thus, U-Safety can extract more effective feature representation of features and achieve better performance than learning method without feature fusion.

Performance of Co-training Learning. We further evaluate the performance of co-training learning progress. We compare the precision and recall of safety index inference with our proposed co-training method, only spatial classifier (SC) and

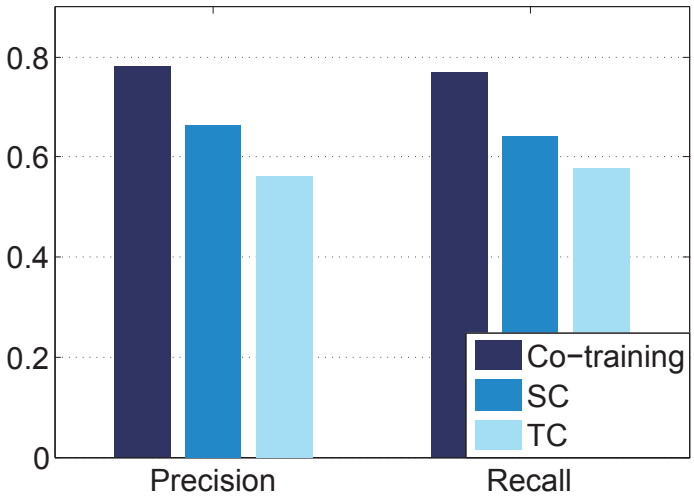


Fig. 5.5: Performance of co-training learning.

only temporal classifier (TC). As revealed in Figure 5.5, the inference of safety index achieves both higher precision and recall with the co-training framework. The precision and recall of safety index inference can reach 78.1% and 77.2% respectively with the co-training learning progress. When we only apply spatial classifier (SC) to infer the safety index, the precision and recall of safety index inference decrease to 66.4% and 64.2% respectively. When we only apply temporal classifier (TC) to infer the safety index, the precision and recall of safety index inference are 56.1% and 57.8% respectively. These results demonstrate the effectiveness of our proposed co-training method. Table 5.4 presents the confusion matrix of U-Safety in inferring safety index in New York City.

Performance of Spatial Classifier. To further evaluate the performance of our proposed system in inferring different safety index levels, we also individually test spatial classifier with only spatially-related features fed. In this experiment, we intentionally constructed our testing data set in which different safety index levels

Table 5.4: Confusion matrix of U-Safety on safety index

Ground Truth	**Predictions**						
	D	**U**	**R**	**M**	**S**		
D	409	75	31	0	0	0.794	**Recall**
U	43	795	127	45	21	0.771	
R	38	62	1106	72	248	0.725	
M	9	34	71	2592	838	0.731	
S	0	0	234	923	5983	0.838	
	0.820	0.823	0.705	0.714	0.844	0.791	
	Precision						

Table 5.5: Confusion matrix of Spatial Classifier

Ground Truth	**Predictions**						
	D	**U**	**R**	**M**	**S**		
D	949	258	86	51	0	0.706	**Recall**
U	232	813	173	83	72	0.592	
R	114	166	1134	186	118	0.660	
M	53	109	266	971	165	0.621	
S	0	68	93	267	1275	0.749	
	0.704	0.575	0.647	0.623	0.782	0.668	
	Precision						

have a relatively balanced distribution. The purpose of doing this is that a balanced data set may produce a more comprehensive evaluation when we conduct the safety index inference. Then we constructed the training data set with a similar distribution of safety index level and an equal size of the data set. Table 5.5 describes the confusion matrix of spatial classifier in safety index level inference.

Performance of Temporal Classifier. In addition, we also individually test temporal classifier with only temporally-related features fed. In this experiment, we use the same training data set and testing data set as in the evaluation of performance of spatial classifier. Table 5.6 describes the confusion matrix of temporal classifier in safety index level inference.

Table 5.6: Confusion matrix of Temporal Classifier

Ground Truth	Predictions						
	D	**U**	**R**	**M**	**S**		
D	782	284	147	97	34	0.582	**Recall**
U	203	797	195	135	43	0.580	
R	82	247	973	233	183	0.566	
M	76	133	274	796	285	0.509	
S	63	102	158	327	1053	0.618	
	0.648	0.510	0.557	0.501	0.659	0.571	
	Precision						

Overall Performance Comparison. In addition, we compare our U-Safety approach with other data analysis methods, including decision tree (DT) [Fong and Weber-Jahnke, 2012], k-nearest neighbors (KNN) [Chen et al., 2013], artificial neural network (ANN) [Sutskever et al., 2014], and support vector machine (SVM) [Huang et al., 2012]. As illustrated in Figure 5.6, U-Safety outperforms other methods in terms of the mean precision and mean recall. The precision and recall of safety index inference can reach 78.1% and 77.2% respectively with our proposed U-Safety approach. When we apply the DT algorithm to infer the safety index, the precision and recall of safety index inference are 48.3% and 47.1% respectively. When the KNN algorithm is utilized to infer the safety index, the precision and recall of safety index inference are 67.3% and 65.9% respectively. And the precision and recall of safety index inference are 58.4% and 57.3% respectively, when the safety index is inferred by the ANN algorithm. Moreover, when we apply the SVM algorithm to conduct the safety index inference, the precision and recall of safety index inference are 53.4% and 55.1% respectively. Thus, the experiment results demonstrate the advantage of our method based on feature fusion and co-training learning.

Visualization. We infer the safety index of each position in urban areas of New

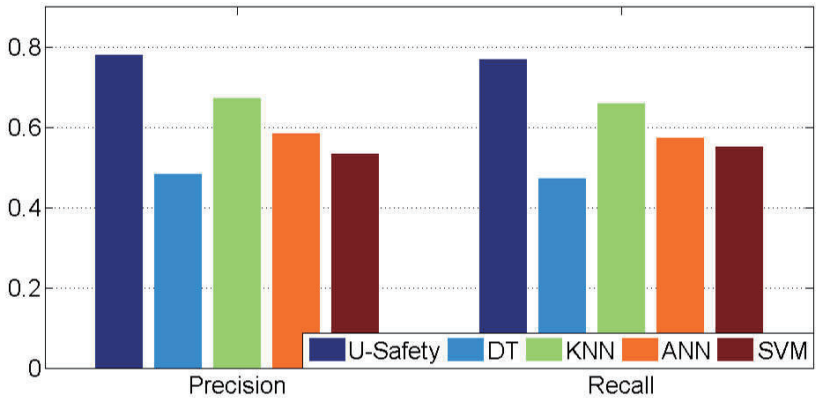
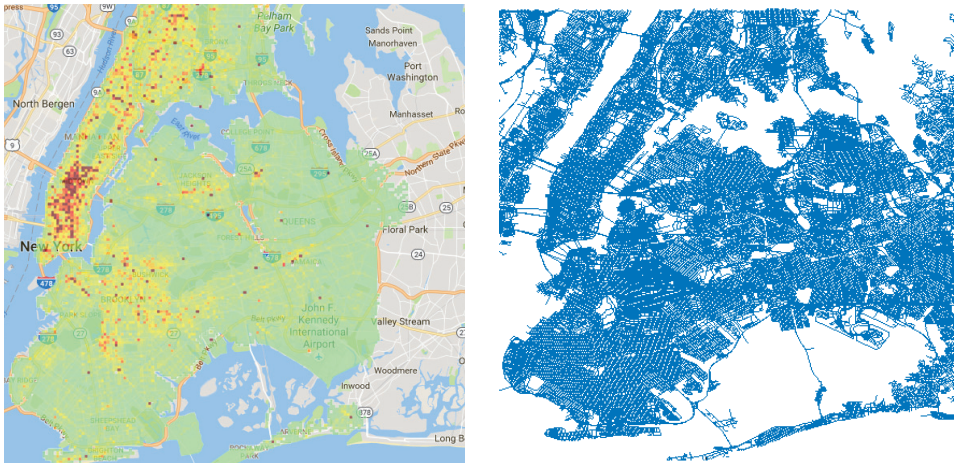


Fig. 5.6: Overall performance comparison of different methods.

York City by utilizing our proposed U-Safety approach. Figure 5.7 (a) presents the visualization of safety index map. The color in each grid denotes the inferred safety index level, as defined in Table 5.1. Figure 5.7 (b) shows the corresponding abstraction of road network of New York City. We believe the visualization can not only benefit urban safety analysis by identifying the areas always having a low safety index level, but also assist people in urban route planning by avoiding relative dangerous areas.

5.5 Chapter Summary

In this chapter, we propose U-Safety, an urban safety analysis system to infer safety index by leveraging multiple cross-domain urban data. Compared with previous works, U-Safety represents the first attempt to cope with urban safety dynamics to build a periodically updated safety index map. To realize this system, we first extract spatially-related and temporally-related features from various urban data. Then, these features are fed into a novel sparse auto-encoder (SAE) with feature correlation constraint to obtain the final discriminative feature representation. Finally, we design a new co-training-based learning method, which consists of two separated classifiers,



(a) Safety index map. (b) Abstraction of road network.

Fig. 5.7: Visualization of safety index inference in New York City.

to calculate safety index accurately. We implement U-Safety and conduct extensive experiments based on real data sources collected in New York City. The evaluation results demonstrate the advantages of U-Safety over other methods.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Mobile sensing is a new technology to construct and update urban digital maps for location based services. Through analyzing multiple sensing data collected from mobile devices, we proposed novel methods to construct indoor floor plan, update outdoor map, and construct urban safety index map.

In the indoor floor plan construction task, we utilize machine learning techniques to build PlanSketcher, a system that enables one user to construct fine-grained and facility-labelled indoor floor plans accurately. First, the proposed system extracts novel integrated features to recognize diverse landmarks. Second, traverse-independent hallway topologies are constructed based on the sensing data, depth data and images through the proposed hallway construction algorithms. Finally, PlanSketcher constructs the room shape and labels recognized facilities in their corresponding positions to generate a complete indoor floor plan. Because PlanSketcher exploits different kinds of data collected from smartphones with new feature extraction method, it can obtain accurate indoor floor plan topology and facility labels. We implement PlanSketcher and conduct extensive experiments in 3 large indoor settings.

The evaluation results illustrate that PlanSketcher outperforms the state-of-the-art methods by showing smaller position and orientation error, more recognized facilities and less energy consumption.

For the outdoor map update, we propose CrowdGIS, an automatic store self-updating system for digital maps that leverages street views and sensing data crowd-sourced from mobile users. We first develop a new weighted artificial neural network to learn the underlying relationship between estimated positions and real positions to localize user's shooting positions. Then, a novel text detection method is designed by considering two valuable features, including the color and texture information of letters. In this way, we can recognize complete store name instead of individual letters as in the previous study. Furthermore, we transfer the shooting position to the location of recognized stores in the map. Finally, CrowdGIS considers three updating categories (replacing, adding, and deleting) to update changed stores in the map based on the kernel density estimate model. We implement CrowdGIS and conduct extensive experiments in a real outdoor region for 1 month. The evaluation results demonstrate that CrowdGIS effectively accommodates store variations and updates stores to maintain an up-to-date map with high accuracy.

Considering urban safety related services, we propose U-Safety, an urban safety analysis system to infer safety index by leveraging multiple cross-domain urban location-based data. We first extract spatially-related and temporally-related features from various urban location-based data, including urban map, housing rent and density, population, positions of police stations, point of interests (POIs), crime event records, and taxi GPS trajectories. Then, these features are fed into a novel sparse auto-encoder (SAE) framework with feature correlation constraint to obtain the final

discriminative feature representation. Finally, we design a new co-training-based learning method, which consists of two separated classifiers, to calculate safety index accurately. We implement U-Safety and conduct extensive experiments by utilizing various real data sources obtained in New York City. The evaluation results demonstrate the advantages of U-Safety over other methods.

6.2 Future Work

As mobile devices are expected to become comparable with personal computers, more effective location based services should be provided to users. We present some future research directions as follows.

Indoor location based augmented reality (AR). As a novel way of presenting information, augmented reality (AR) enables people to interact with the physical world in a direct and intuitive way. Indoor AR service, as a new attractive location based service, is becoming more popular when the indoor location of a user can be accurately determined. Traditional AR systems utilize image processing techniques to determine the position of information displayed on the smartphone screen. These vision-based methods need to continuously track predefined markers within a short distance and consume much energy of smartphone, which greatly degrade user experience. Thus, a novel scalable indoor AR framework based on localization should be proposed.

Dynamic road safety index prediction for vehicles. Information about vehicle safety, such as the driving safety status and the road safety index, is of great importance to protect humans and support safe driving route planning. Despite some research on driving safety analysis, the accuracy and granularity of driving safety

assessment are both very limited. And the problem of precisely and dynamically predicting road safety index throughout a city has not been sufficiently studied and remains open. With the proliferation of sensor-equipped vehicles and smart devices, a huge amount of mobile sensing data provide an opportunity to conduct vehicle safety analysis. Thus, how to develop an effective system to dynamically predict real-time road safety index remains an unsolved problem.

Charging stations placement for electric vehicles. The rapid progress of urbanization is consuming more and more energy, which calls for technologies to sense city-scale energy cost, improve energy infrastructures, and finally reduce energy consumption. With an increase in electrification of vehicles, electric vehicles will become more popular in the near future. Large amounts of new charging stations need to be deployed in different locations in the city. However, how to select appropriate positions to provide charging service to realize the optimal utilization of charging stations is still an open issue. A promising method is mining mobile sensing data collected from sensor-equipped vehicles and smart devices to identify locations of interest for the deployment. Thus, we will attempt to investigate the possibility of predicting electric vehicles charging demands at different locations in the city.

Bibliography

Monsoon power monitor. http://www.msoon.com/LabEquipment/PowerMonitor/.

Opencv. http://opencv.org/.

Community crime map. https://communitycrimemap.com/#nwrButton.

Moustafa Alzantot and Moustafa Youssef. Crowdinside: automatic construction of indoor floorplans. In *Proc. of ACM SIGSPATIAL*, 2012.

Galen Andrew, Raman Arora, Jeff A Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *Proc. of ACM ICML*, 2013.

Sean M Arietta, Alexei A Efros, Ravi Ramamoorthi, and Maneesh Agrawala. City forensics: Using visual elements to predict non-visual city attributes. *IEEE transactions on visualization and computer graphics*, 20(12):2624–2633, 2014.

Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of ACM COLT*, 1998.

Zdravko I Botev, Joseph F Grotowski, Dirk P Kroese, et al. Kernel density estimation via diffusion. *The Annals of Statistics*, 38(5):2916–2957, 2010.

Li-Juan Cao and Francis Eng Hock Tay. Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on neural networks*, 14(6):1506–1518, 2003.

Gert Cauwenberghs and Tomaso Poggio. Incremental and decremental support vector machine learning. In *Proc. of NIPS*, 2000.

Qifeng Chen, Dingzeyu Li, and Chi-Keung Tang. Knn matting. *IEEE transactions on pattern analysis and machine intelligence*, 35(9):2175–2188, 2013.

Si Chen, Muyuan Li, Kui Ren, Xinwen Fu, and Chunming Qiao. Rise of the indoor crowd: Reconstruction of building interior view via mobile crowdsourcing. In *Proc. of ACM SenSys*, 2015a.

Si Chen, Muyuan Li, Kui Ren, and Chunming Qiao. Crowd map: Accurate reconstruction of indoor floor plans from crowdsourced sensor-rich videos. In *Proc. of IEEE ICDCS*, 2015b.

Delphine Christin, Christian Roßkopf, and Matthias Hollick. usafe: A privacy-aware and participative mobile application for citizen safety in urban environments. *Pervasive and Mobile Computing*, 9(5):695–707, 2013.

Javier Civera, Matei Ciocarlie, Alper Aydemir, Kostas Bekris, and Sanjay Sarma. Guest editorial special issue on cloud robotics and automation. *IEEE Transactions on Automation Science and Engineering*, 12(2):396–397, 2015.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 5(4):455–455, 1992.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. of IEEE CVPR*, 2005.

Subhankar Dhar and Upkar Varshney. Challenges and business models for mobile location-based services and advertising. *Communications of the ACM*, 54(5):121–128, 2011.

Stephan Dreiseitl and Lucila Ohno-Machado. Logistic regression and artificial neural network classification models: a methodology review. *Journal of biomedical informatics*, 35(5):352–359, 2002.

Li Fei-Fei and Pietro Perona. A bayesian hierarchical model for learning natural scene categories. In *Proc. of IEEE CVPR*, 2005.

Pui Kuen Fong and Jens H Weber-Jahnke. Privacy preserving decision tree learning using unrealized data sets. *IEEE Transactions on knowledge and Data Engineering*, 24(2):353–364, 2012.

Raghu K Ganti, Fan Ye, and Hui Lei. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, 49(11):32–39, 2011.

Ruipeng Gao, Mingmin Zhao, Tao Ye, Fan Ye, Yizhou Wang, Kaigui Bian, Tao Wang, and Xiaoming Li. Jigsaw: Indoor floor plan reconstruction via mobile crowdsensing. In *Proc. of ACM MobiCom*, 2014.

Ruipeng Gao, Yang Tian, Fan Ye, Guojie Luo, Kaigui Bian, Yizhou Wang, Tao Wang, and Xiaoming Li. Sextant: Towards ubiquitous indoor localization service by photo-taking of the environment. *IEEE Transactions on Mobile Computing*, 15 (2):460–474, 2016.

Shang Gao, Zhe Peng, Bin Xiao, Qingjun Xiao, and Yubo Song. Scop: Smartphone energy saving by merging push services in fog computing. In *Proc. of IEEE IWQoS*, 2017.

Muriel Gevrey, Ioannis Dimopoulos, and Sovan Lek. Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecological modelling*, 160(3):249–264, 2003.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. of IEEE CVPR*, 2014.

Abhishek Goswami, Luis E Ortiz, and Samir R Das. Wigem: a learning-based approach for indoor localization. In *Proc. of ACM CoNEXT*, 2011.

Xiaonan Guo, Eddie CL Chan, Ce Liu, Kaishun Wu, Siyuan Liu, and Lionel M Ni. Shopprofiler: Profiling shops with crowdsourcing data. In *Proc. of IEEE INFOCOM*, 2014.

David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664, 2004.

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.

Guang-Bin Huang, Hongming Zhou, Xiaojian Ding, and Rui Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, 2012.

Anil K Jain, Yu Zhong, and Sridhar Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18 (3):267–278, 1996.

Puneet Jain, Justin Manweiler, Arup Acharya, and Kirk Beaty. Focus: clustering crowdsourced videos by line-of-sight. In *Proc. of ACM SenSys*, 2013.

Yifei Jiang, Yun Xiang, Xin Pan, Kun Li, Qin Lv, Robert P Dick, Li Shang, and Michael Hannigan. Hallway based automatic indoor floorplan construction using room fingerprints. In *Proc. of ACM Ubicomp*, 2013.

Wazir Zada Khan, Yang Xiang, Mohammed Y Aalsalem, and Quratulain Arshad. Mobile phone sensing systems: A survey. *IEEE Communications Surveys & Tutorials*, 15(1):402–427, 2013.

Aditya Khosla, Byoungkwon An, Joseph J Lim, and Antonio Torralba. Looking beyond the visible scene. In *Proc. of IEEE CVPR*, 2014.

Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Comprehensive review of neural network-based prediction intervals and new advances. *IEEE Transactions on neural networks*, 22(9):1341–1356, 2011.

Neil E Klepeis, William C Nelson, Wayne R Ott, John P Robinson, Andy M Tsang, Paul Switzer, Joseph V Behar, Stephen C Hern, and William H Engelmann. The national human activity pattern survey (nhaps): a resource for assessing exposure to environmental pollutants. *Journal of Exposure Science and Environmental Epidemiology*, 11(3):231, 2001.

Nicholas D Lane, Emiliano Miluzzo, Hong Lu, Daniel Peebles, Tanzeem Choudhury, and Andrew T Campbell. A survey of mobile phone sensing. *IEEE Communications magazine*, 48(9), 2010.

David C Lee, Martial Hebert, and Takeo Kanade. Geometric reasoning for single image structure recovery. In *Proc. of IEEE CVPR*, 2009.

Jiwei Li, Zhe Peng, Shang Gao, Bin Xiao, and Henry Chan. Smartphone-assisted energy efficient data communication for wearable devices. *Computer Communications*, 105:33–43, 2017.

Mo Li, Pengfei Zhou, Yuanqing Zheng, Zhenjiang Li, and Guobin Shen. Iodetector: A generic service for indoor/outdoor detection. *ACM Transactions on Sensor Networks (TOSN)*, 11(2):28, 2015.

Robert LiKamWa, Bodhi Priyantha, Matthai Philipose, Lin Zhong, and Paramvir Bahl. Energy characterization and optimization of image sensing toward continuous mobile vision. In *Proc. of ACM MobiSys*, 2013.

Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C Weng. A note on platts probabilistic outputs for support vector machines. *Machine learning*, 68(3):267–276, 2007.

Fangming Liu, Peng Shu, Hai Jin, Linjie Ding, Jie Yu, Di Niu, and Bo Li. Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications. *Wireless Communications, IEEE*, 20(3):14–22, 2013.

Fangming Liu, Peng Shu, and John Lui. Appatp: An energy conserving adaptive mobile-cloud transmission protocol. *Computers, IEEE Transactions on*, 64(11): 3051–3063, 2015.

Hongbo Liu, Yu Gan, Jie Yang, Simon Sidhom, Yan Wang, Yingying Chen, and Fan Ye. Push the limit of wifi based localization for smartphones. In *Proc. of ACM MobiCom*, 2012a.

Kuien Liu, Yaguang Li, Fengcheng He, Jiajie Xu, and Zhiming Ding. Effective map-matching on the most simplified road network. In *Proc. of ACM SIGSPATIAL GIS*, 2012b.

Xuan Liu, Shigeng Zhang, Bin Xiao, and Kai Bu. Flexible and time-efficient tag scanning with handheld readers. *IEEE Transactions on Mobile Computing*, 15(4): 840–852, 2016.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

Bangalore S Manjunath and Wei-Ying Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, 1996.

Justin Gregory Manweiler, Puneet Jain, and Romit Roy Choudhury. Satellites in our pockets: an object positioning system using smartphones. In *Proc. of ACM MobiSys*, 2012.

Sorin Matei, Sandra J Ball-Rokeach, and Jack Linchuan Qiu. Fear and misperception of los angeles urban space a spatial-statistical study of communication-shaped mental maps. *Communication Research*, 28(4):429–463, 2001.

Gajamohan Mohanarajah, Vladyslav Usenko, Mayank Singh, Raffaello D'Andrea, and Markus Waibel. Cloud-based collaborative 3d mapping in real-time with low-cost robots. *IEEE Transactions on Automation Science and Engineering*, 12(2): 423–431, 2015.

Marjan Momtazpour, Patrick Butler, M Shahriar Hossain, Mohammad C Bozchalui, Naren Ramakrishnan, and Ratnesh Sharma. Coordinated clustering algorithms to support charging infrastructure design for electric vehicles. In *Proc. of ACM KDD*, 2012.

Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 2017.

Sarfraz Nawaz, Christos Efstratiou, and Cecilia Mascolo. Parksense: A smartphone based sensing system for on-street parking. In *Proc. of ACM MobiCom*, 2013.

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *Proc. of ACM ICML*, 2011.

Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and applicability of co-training. In *Proc. of ACM CIKM*, 2000.

Vicente Ordonez and Tamara L Berg. Learning high-level judgments of urban perception. In *Proc. of ACCV*. Springer, 2014.

Greg Pass, Ramin Zabih, and Justin Miller. Comparing images using color coherence vectors. In *Proc. of ACM MULTIMEDIA*, 1997.

VV Phansalkar and PS Sastry. Analysis of the back-propagation algorithm with momentum. *IEEE Transactions on Neural Networks*, 5(3):505–506, 1994.

Damian Philipp, Patrick Baier, Christoph Dibak, Frank Durr, Kurt Rothermel, Susanne Becker, Michael Peter, and Dieter Fritsch. Mapgenie: Grammar-enhanced indoor map construction from crowd-sourced data. In *Proc. of IEEE Percom*, 2014.

John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

Anshul Rai, Krishna Kant Chintalapudi, Venkata N Padmanabhan, and Rijurekha Sen. Zee: zero-effort crowdsourcing for indoor localization. In *Proc. of ACM MobiCom*, 2012.

Jason Rife, Sam Pullen, Boris Pervan, and Per Enge. Paired overbounding and application to gps augmentation. In *Proc. of IEEE PLANS*, 2004.

Deborah Salon. Neighborhoods, cars, and commuting in new york city: A discrete choice approach. *Transportation Research Part A: Policy and Practice*, 43(2):180–196, 2009.

Longfei Shangguan, Zimu Zhou, Zheng Yang, Kebin Liu, Zhenjiang Li, Xibin Zhao, and Yunhao Liu. Towards accurate object localization with smartphones. *IEEE Transactions on Parallel and Distributed Systems*, 25(10):2731–2742, 2014.

Guobin Shen, Zhuo Chen, Peichao Zhang, Thomas Moscibroda, and Yongguang Zhang. Walkie-markie: indoor pathway mapping made easy. In *Proc. of USENIX NSDI*, 2013.

Haichen Shen, Aruna Balasubramanian, Eric Yuan, Anthony LaMarca, and David Wetherall. Improving power effciency using sensor hubs without re-coding mobile apps. In *Technical Report of University of Washington*, 2014.

Hoo-Chang Shin, Matthew R Orton, David J Collins, Simon J Doran, and Martin O Leach. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1930–1943, 2013.

Peng Shu, Fangming Liu, Hai Jin, Min Chen, Feng Wen, and Yupeng Qu. etime: energy-efficient transmission between cloud and mobile devices. In *Proc. of IEEE INFOCOM*, 2013.

Yuanchao Shu, Kang G Shin, Tian He, and Jiming Chen. Last-mile navigation using smartphones. In *Proc. of ACM MobiCom*, 2015.

Pieter Simoens, Yu Xiao, Padmanabhan Pillai, Zhuo Chen, Kiryong Ha, and Mahadev Satyanarayanan. Scalable crowd-sourcing of video from mobile devices. In *Proc. of ACM MobiSys*, 2013.

Nitish Srivastava and Ruslan R Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Proc. of NIPS*, 2012.

Peter Sturm. Pinhole camera model. In *Computer Vision*, pages 610–613. Springer, 2014.

Liang Sun, Shuiwang Ji, and Jieping Ye. Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):194–200, 2011.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Proc. of NIPS*, 2014.

Renran Tian, Lingxi Li, Mingye Chen, Yaobin Chen, and Gerald J Witt. Studying the effects of driver distraction and traffic density on the probability of crash and near-crash events in naturalistic driving environment. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1547–1555, 2013.

Martin Traunmueller, Paul Marshall, and Licia Capra. ... when youre a stranger: Evaluating safety perceptions of (un) familiar urban places. In *Proc. of ACM UrbIoT*, 2016.

Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

Carlos A Vanegas, Daniel G Aliaga, and Bedrich Benes. Automatic extraction of manhattan-world building masses from 3d laser range scans. *IEEE transactions on visualization and computer graphics*, 18(10):1627–1637, 2012.

He Wang, Souvik Sen, Ahmed Elgohary, Moustafa Farid, Moustafa Youssef, and Romit Roy Choudhury. No need to war-drive: unsupervised indoor localization. In *Proc. of ACM MobiSys*, 2012.

Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *Proc. of IEEE CVPR*, 2010.

Yilun Wang, Yu Zheng, and Yexiang Xue. Travel time estimation of a path using sparse trajectories. In *Proc. of ACM KDD*, 2014.

Yin Wang, Xuemei Liu, Hong Wei, George Forman, Chao Chen, and Yanmin Zhu. Crowdatlas: self-updating maps for cloud and personal use. In *Proc. of ACM Mobisys*, 2013.

Y Wen, X Tian, X Wang, and S Lu. Fundamental limits of rss fingerprinting based indoor localization. In *Proc. of IEEE INFOCOM*, 2015.

Chenshu Wu, Zheng Yang, Chaowei Xiao, Chaofan Yang, Yunhao Liu, and Mingyan Liu. Static power of mobile devices: Self-updating radio maps for wireless indoor localization. In *Proc. of IEEE INFOCOM*, 2015a.

Chenshu Wu, Zheng Yang, Yu Xu, Yiyang Zhao, and Yunhao Liu. Human mobility enhances global positioning accuracy for mobile phone localization. *Parallel and Distributed Systems, IEEE Transactions on*, 26(1):131–141, 2015b.

Fan Yang, Huchuan Lu, and Yen-Wei Chen. Human tracking by multiple kernel boosting with locality affinity constraints. In *Computer Vision–ACCV 2010*, pages 39–50. Springer, 2011.

Fan Yang, Qiang Zhai, Guoxing Chen, Adam C Champion, Junda Zhu, and Dong Xuan. Flash-loc: Flashing mobile phones for accurate indoor localization. In *Proc. of IEEE INFOCOM*, 2016.

Zheng Yang, Chenshu Wu, and Yunhao Liu. Locating in fingerprint space: wireless indoor localization with little human intervention. In *Proc. of ACM MobiCom*, 2012.

Cong Yao, Xiang Bai, and Wenyu Liu. A unified framework for multioriented text detection and recognition. *IEEE Transactions on Image Processing*, 23(11):4737–4749, 2014.

Jie Yin, Qiang Yang, and Lionel M Ni. Learning adaptive temporal radio maps for signal-strength-based location estimation. *IEEE Transactions on Mobile Computing*, 7(7):869–883, 2008.

Yixuan Yuan and Max Q-H Meng. Deep learning for polyp recognition in wireless capsule endoscopy images. *Medical Physics*, 44(4):1379–1389, 2017.

Yixuan Yuan, Jiaole Wang, Baopu Li, and Max Q-H Meng. Saliency based ulcer detection for wireless capsule endoscopy diagnosis. *IEEE transactions on medical imaging*, 34(10):2046–2057, 2015.

Yixuan Yuan, Baopu Li, and Max Q-H Meng. Wce abnormality detection based on saliency and adaptive locality-constrained linear coding. *IEEE Transactions on Automation Science and Engineering*, 14(1):149–159, 2017a.

Yixuan Yuan, Xiwen Yao, Junwei Han, Lei Guo, and Max Q-H Meng. Discriminative joint-feature topic model with dual constraints for wce classification. *IEEE Transactions on Cybernetics*, 2017b.

Özgür Yürür, Chi Harold Liu, Zhengguo Sheng, Victor CM Leung, Wilfrido Moreno, and Kin K Leung. Context-awareness for mobile sensing: A survey and future directions. *IEEE Communications Surveys & Tutorials*, 18(1):68–93, 2016.

Lan Zhang, Kebin Liu, Yonghang Jiang, Xiang-Yang Li, Yunhao Liu, and Panlong Yang. Montage: Combine frames with movement continuity for realtime multi-user tracking. In *Proc. of IEEE INFOCOM*, 2014.

Lei Zhang, Yongdong Zhang, Jinhu Tang, Ke Lu, and Qi Tian. Binary code ranking with weighted hamming distance. In *Proc. of IEEE CVPR*, 2013.

Tan Zhang, Xian Zhang, Fangming Liu, Hongkun Leng, Qian Yu, and Guanfeng Liang. etrain: Making wasted energy useful by utilizing heartbeats for mobile data transmissions. In *Proc. of IEEE ICDCS*, 2015.

Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *Proc. of ACM SIGMOD*, 1996.

Zengbin Zhang, Xia Zhou, Weile Zhang, Yuanyang Zhang, Gang Wang, Ben Y Zhao, and Haitao Zheng. I am the antenna: accurate outdoor ap location using smartphones. In *Proc. of ACM MobiCom*, 2011.

Vincent W Zheng, Yu Zheng, Xing Xie, and Qiang Yang. Collaborative location and activity recommendations with gps history data. In *Proc. of ACM WWW*, 2010.

Yu Zheng, Furui Liu, and Hsun-Ping Hsieh. U-air: when urban air quality inference meets big data. In *Proc. of ACM SIGKDD*, 2013.

Yuanqing Zheng, Guobin Shen, Liqun Li, Chunshui Zhao, Mo Li, and Feng Zhao. Travi-navi: Self-deployable indoor navigation system. In *Proc. of ACM MobiCom*, 2014.

Pengfei Zhou, Mo Li, and Guobin Shen. Use it free: Instantly knowing your phone attitude. In *Proc. of ACM MobiCom*, 2014.