Vision-based Assistive Indoor Localization

Feng HU

A dissertation thesis submitted to
the Graduate Faculty in Computer Science in partial fulfillment of the
requirements for the degree of Doctor of Philosophy,
The City University of New York.

December 2016

This manuscript has been read and accepted by the Graduate Faculty in Computer Sciencein satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Professor Zhigang Zhu

Date	Chair of Examining Committee

Professor Robert Haralick

Date	Executive Officer

Supervisory Committee

Professor Zhigang Zhu (The Graduate Center and City College, CUNY)

Professor Jizhong Xiao (The Graduate Center and City College, CUNY)

Professor Jianting Zhang (The Graduate Center and City College, CUNY)

Dr. Ying Li (Thomas J. Watson Research Center, IBM)

The City University of New York

Abstract

Vision-based Assistive Indoor Localization

by

Feng HU

Advisor: Professor Zhigang Zhu

An indoor localization system is of significant importance to the visually impaired in their daily lives by helping them localize themselves and further navigate an indoor environment. In this thesis, a vision-based indoor localization solution is proposed and studied with algorithms and their implementations by maximizing the usage of the visual information surrounding the users for an optimal localization from multiple stages. The contributions of the work include the following: (1) Novel combinations of a daily-used smart phone with a low-cost lens (GoPano) are used to provide an economic, portable, and robust indoor localization service for visually impaired people. (2) New omnidirectional features (omni-features) extracted from 360 degrees field-of-view images are proposed to represent visual landmarks of indoor positions, and then used as on-line query keys when a user asks for localization services. (3) A scalable and light-weight computation and storage solution is implemented by transferring big database storage and computational heavy querying procedure to the cloud. (4) Real-time query performance of 14 fps is achieved with a Wi-Fi connection by identifying and implementing both data and task parallelism using many-core NVIDIA GPUs. (5) Refine localization via 2D-to-3D and 3D-to-3D geometric matching and automatic path planning for efficient environmental modeling by utilizing architecture AutoCAD floor plans.

This dissertation first provides a description of assistive indoor localization problem with its detailed connotations as well as overall methodology. Then related work in indoor localization and automatic path planing for environmental modeling is surveyed. After that, the framework of omnidirectional-vision-based indoor assistive localization is introduced. This is followed by multiple refine localization strategies such as 2D-to-3D and 3D-to-3D geometric

matching approaches. Finally, conclusions and a few promising future research directions are provided.

Acknowledgements

This work is supported by the U.S. National Science Foundation (NSF) through Award # EFRI-1137172 and the CUNY Graduate Center Science Fellowship (2012-2014). This work has also been supported by CUNY Graduate Center Doctoral Student Research Grant.

Part of Chapter 3 is published, collaborating with H. Tang, N. Tsering, in "Automatic Pre-Journey Indoor Map Generation Using AutoCAD Floor Plan", *Journal on Technology & Persons with Disabilities*, San Diego, CA, U.S.A. 2016. Part of Chapter 4 is published, collaborating with Z. Zhu, J. Mejia, H. Tang, and J. Zhang, in "Mobile Panoramic Vision for Assisting Blind via Indexing and Localization" *2014 European Conference on Computer Vision Workshops*, Zurich, Sep. 6-12, 2014, and "Real-time indoor assistive localization with mobile omnidirectional vision and cloud GPU acceleration", *Journal of Real-Time Image Processing* (pending), 2016. Part of Chapter 5 is published, collaborating with H. Tang, N. Tsering and Z. Zhu, in "RGB-D Sensor Based Indoor Localization for the Visually Impaired", *Journal on Technology & Persons with Disabilities*, San Diego, CA, U.S.A. 2016.

I want to thank my advisor, Prof. Zhigang Zhu, for his constant support from all perspectives, for his insightful suggestions in selecting research topics and conducting research, and for his warmth encouragement, as well as mentoring me on how to mentor and teaching me on how to teach. His research, mentoring, teaching and collaboration styles will significantly influence my life-long career development.

I want to thank my committee members, Prof. Jizhong Xiao, Prof. Jianting Zhang, and Dr. Ying Li, for their valuable and insightful comments and feedback in my dissertation work and career development. I also want to thank Prof. Jianting Zhang for providing the Project Tango tablet and high perfornmance computing resources for conducting my research.

I also want to thank my colleagues (former and current): Dr. Hao Tang, Dr. Edgardo Molina, Dr. Wai L. Khoo, Dr. Tao Wang, Mr. Martin Goldberg, Ms. Farnaz Abtahi, Mr. Wei Li, and Mr. Greg Olmschenk, for their support, friendship, and collaborations, allowing me to finish my research in a comfortable environment.

I would like to thank Ms. Barbara Campbell and her colleagues of the New York State Commission for the Blind (NYSCB) for their comments and suggestions, thank Mr. Bing Li and Mr. Huang Zou for their assistance in the path planning for data collection work.

I would like to thank all the students I have mentored/collaborated, including: Jeury Mejia, Norbu Tsering, Kenichi Yamamoto and many others, for helping me in various projects and teaching me how to be a better mentor.

I would also like to thanks my friends: Dr. Xin Gong, Dr. Bing Tang, Dr. Lei Zheng, Dr. Chucai Yi, and many others for sharing a great life in NYC.

Finally, I would like to thank all the other people unnamed who have helped me for turning my research and thesis into reality.

To my dear parents and family members. To my girlfriend/wife (who has not shown up yet), without whom I can finish my thesis on time.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

An indoor localization system is of significant importance to the visually impaired in their daily lives if it can help them localize themselves and further navigate unfamiliar indoor environments. There are 285 million visually impaired people in the world according to the World Health Organization (last time accessed: Nov. 2016), among whom 39 million are blind [1]. Compared to sighted people, it is much harder for visually impaired people to navigate indoor environments. Nowadays, too many buildings are also unfortunately mainly designed and built for sighted people; therefore, navigational tasks and functions that sighted people take for granted could be huge problems to visually impaired people. Despite a large amount of research have been carried out for robot navigation in the robotics community, and several assistive systems are designed for blind people, efficient and effective portable solutions for visually impaired people are not yet available. In this thesis, a robust localization and navigation solution is provided for visually impaired people using portable devices, e.g., GoPano lens and iPhone[2], Google Glass[3] or Project Tango Tablet[4].

[1] http://www.who.int/mediacentre/factsheets/fs282/en/
[2] http://eyesee360.com
[3] https://developers.google.com/glass/
[4] http://get.google.com/tango/

1.1 Problem statement

When it comes to the task of indoor localization, normal sighted people usually localize themselves by looking at the surrounding visual information to identify their current locations, e.g., on a floor plan, and then determine the appropriate paths to their desired locations. Due to the lack of visual input, visually impaired people face great challenges in the localization and navigation tasks when entering into a new indoor environment.

With the advances in image processing and computer vision techniques, and the emergence of ubiquitous of portable imaging systems and mobile computing devices, we formalize our problem statement in this thesis as:

How can we make full use of the already existing visual information around a visually impaired person, and nonintrusive portable devices available to them, to provide a real-time, accurate and robust indoor localization service?

In the following, the connotations of this problem statement will be elaborated in detail by describing the concepts, the designs and implementations that could lead to the solutions to answer this problem.

(1) Indoor localization

Indoor localization is to use one or more sensors such as cameras, magnetic sensors, IMUs, RGBD sensors, etc., to automatically determine the location of a robot or a person in real-time in an indoor environment. This thesis mainly focuses on the indoor localization solutions with visual sensors, including normal camera, camera with omnidirectional lens, or 3D sensors, for assisting visually impaired people with performing indoor navigation tasks.

A successful assistive indoor localization system will have great social and economic impact to the visually impaired community and the society. The usual way for visually impaired people to navigate indoor environments is by memorization, which means, by remembering the already traveled route and recalling it when they come to the same place the second

time [75]. If we review this psychological process, we will find a lot of efforts are involved in the mental task. For example, they have to precisely remember turn-by-turn information and all the important points of interests (i.e., landmarks) along the routes. Without these landmarks, they lose the waypoints to help themselves to judge whether they are on the correct path or not. Therefore, it would be very meaningful to create and build assistive indoor navigation systems for the visually impaired to relieve them from such memory burdens, and free their mind to deal other important issues in their daily lives.

(2) Why visual information?

Visual information is widely used by human beings, as well as many other species, to perceive their environments and carry out daily tasks. As an important perspective of artificial intelligence, visual information processing is the visual reasoning skill that enables us to process and interpret meaning from visual information that we gain through our eyesight.

Visual perception plays a big role in our daily life, but probably because of the ease with which we rely on the perception, we tend to overlook the complexity behind it. Understanding how we interpret what we see can help us design and organize our visual information and inspire many useful applications for the real world.

Visual information based localization is one of such applications, and is now attracting more and more interest in both the computer vision community and the robotics community, mainly to achieve real-time and accurate localization [36][31][33].

Visual information encode very rich knowledge about the indoor environment. First, it includes all the visible physical objects existing in the environments, for example, the doors, windows, notice boards. Even though the general problem of object recognition is not fully solved yet, there are sufficient techniques we can utilize to explore these visual information, for example, by extracting primitive geometric elements, such as vertical lines, to differentiate environmental landmarks for an accurate localization. Second, visual information can provide structure information, for example 3D model, of an environment. These structure

information can be used a global matching base for a later localization query from a user providing just an image, an image sequence or a local 3D model.

(3) Nonintrusive portable devices

Some researchers designed special robots or other systems for helping visually impaired people [103], however, according to previous research and discussion with many visually impaired individuals, the majority of them do not want to carry extra devices [42]. There are a couple of reasons. First, they might already have several devices to hold every day (e.g. white canes and braille notetakers). Second, most of them do not want to use systems that attract extra attentions to them and distinguish themselves from normal sighted people especially in the public space. Third, usability study on assistive technology shows that new functionality provided on the existing devices (e.g. daily-used smart phones) or hands-free portable devices, such as wearable glasses, are of their preference [58].

The most common portable nonintrusive devices include iOS based and Android based smart phones, tablets, and wearable glasses. These devices are used daily by not only visually impaired, but also normal sighted people. They are also becoming more and more powerful both in terms of computation and storage, and in providing high-quality visual sensors, such as on-board cameras, for sensing the environment.

In this thesis, three popular or potentially promising devices are utilized, GoPano plus iPhone, Google Glass and Project Tango tablet, for their unique functionality provided. GoPano and iPhone combined together provides ultra-wide Field of View (FoV) images; Google Glass provides very natural and convenient user interface; and Project Tango device has fast direct 3D sensing units. Even though it seems that we need multiple devices to make usage of their unique functionality, this is likely not necessary in the future devices, as we see more and more companies are trying to embedded all these features together into their latest products. For example, iPhone 7 plus [5] are integrating wide Field of View camera as

[5] http://www.apple.com/iphone-7/specs/

a built-in camera, and Lenovo Phab phone[6] integrates 3D sensors into their smart phone. In the future, there would be devices that have all the features (portable, wide FOV, 3D sensing enable, natural user interface, etc.) built-in, and we can make use of it for an integrated accurate and convenient localization solution for visually impaired people.

(4) Real-time performance

For localization applications, especially assistive localization for the blind, real-time performance is one of the most critical requirements, as the users usually will lose patience if the system is too slow. There are many challenges in designing and implementing a real-time system. First, since the actual devices the visually impaired used are portable devices, they have limited computation power compared to desktops or computer clusters. Second, as the environment area size scales up, devices such as smart phones do not have sufficient memory space to store such large environmental database.

To ensure a real-time performance, two strategies are involved in the design and implementation process of our thesis work. First, a client-server architecture is adopted to transfer the storage and computation load from the smart phone client, which has limited resources, to the server side, which theoretically has unlimited storage and computation power. Second, GPU acceleration mechanism is utilized to further improve the time performance especially when database size is large.

(5) Levels of accuracy

Accuracy is one of the most important indicators for a useful localization system. Localization result is critical for any further navigation instructions, so an inaccurate localization will make the entire navigation system dysfunctional. However, indoor localization accuracy standards have not been clearly defined by researchers or industries yet. Even though extreme high accuracy, e.g. up to millimeter level, is not necessary for localizing a person for navigation purpose, the higher accuracy, the better. A suggested industrial-level accuracy is

[6] http://shop.lenovo.com/us/en/tango/

2 meters [7], within which a user can differentiate the doors of two nearby offices, detect the floors on which the user is using, or find a car in a parking lot.

There are also a few challenges in the visual information indexing-and-retrieving based indoor localization for achieving high accuracy. First, repetition of the scenes. If multiple scenes are alike, it is hard for the system to distinguish one location from the other based on just the discrete images obtained at these locations. Second, limited Field of View of the cameras. Even though scenes are different with each other, if only a portion of the entire scene is used to extract scene representation features, the representation power of these features will be damaged. Third, trade-off between storage consumption and accuracy. To reduce the storage and computation load, dimensions of the scene features are usually reduced, for example, using Principle Component Analysis (PCA) [40], different features therefore may appear to be similar, and thus influence the accuracy of the localization system.

(6) Robustness

The system shall be able to withstand or overcome many kinds of challenges and adversity. Note that all the information used in vision-based localization are originally from the lights of the environment that are then accumulated by the imaging system. If this imaging procedure is interfered, for example, because of illumination change, occlusion, distortion, etc., the same scene will generate different visual representations, which makes the system unstable.

There are multiple ways to increase the robustness of a system, and some uniqueness of the assistive indoor localization problem can be used as the assumptions. For example, the visually impaired people usually walk slowly and smoothly without dramatic location changes. If there is a localization result significant far away from previous obtained result, e.g. result obtained 5 seconds ago, it could be treated as an unstable result and can be removed.

[7] http://spreo.co/

1.2 Methodology

Depending on the form of localization result we needed, utilizing visual information for the assistive localization can be divided into two categories: metric localization and topological localization. Metric localization provides detailed coordinates of the user (or the sensor) in a given pre-defined global two-dimensional (2D) or three-dimensional (3D) coordinate system. Topological localization does not provide absolute coordinates measurement within any coordinate system, but divides the location results into discrete classes, and therefore provides discrete location class name for any input visual information. In this thesis, we are only focusing on metric localization since it provides more accuracy results and is widely needed in visually impaired people's daily life.

There are many different methods which can be used for visual information based indoor localization, but if we extract their common idea and analyze them from the methodology perspective of view, locating any user (or sensor) using visual information has the paradigm of *computing newly obtained visual information against the pre-built surrounding visual model*.

In this section, instead of discussing any specific method, we in general discuss the methodology behind our research work. We divide the vision based indoor localization task into three components: environmental modeling, localization algorithms, and assistive user interfaces.

1.2.1 Environmental modeling

Representing environment is one of the fundamental components for a visual information based indoor localization systems. The physical space we live in is three dimensional, therefore we can use 3D representations, for example, 3D point cloud, to represent the environments. We can, however, also represent the 3D environment by projecting it into 2D space, and use 2D representations, for example images, to organize environment scenes.

In this subsection, we will discuss two approaches in environmental modeling: omnidirectional imaging based environmental modeling and 3D structure based environmental modeling. The first approach belongs to the 2D category, but utilizes wide field of view images, and thus is more powerful than normal field of view representation. We divide the second approach into two types, sparse 3D points based reconstruction using Structure from Motion technique, and dense 3D points based reconstruction using direct sensing.

(1) 2D omnidirectional imaging based environmental modeling

2D image based environmental modeling registers 2D image information—usually in terms of unique 2D features or feature combinations—and the physical coordinate of the space together, then when another occurrence of these unique features appears, we infer the corresponding location by searching the previous feature-coordinate mapping pairs.

This modeling problem can therefore be converted into feature-coordinate indexing or mapping problem. In general, it includes two steps: (1) acquiring useful images or features, and (2) relating them to the physical locations in the space. To reduce the database scale without losing much accuracy, key frame extraction techniques are usually used to minimize the memory and computation needed by the features in the modeling process.

The indexing procedure is conducted in the modeling stage before the system is delivered to real users, which means that the system developer has to construct the mapping for the area they want to cover. The modeling procedure is, however, only needed once and is offline.

A common way to relate images and the physical world is to use floor plans. For example, we can use the corner of one floor of a building as the origin of the world coordinate system, and all the other image features are registered in this coordinate system afterwards in indoor localization applications.

One advantage of using a global coordinate system is that it can utilize the coordinates of some known position, e.g. the entrance of a building, as the initial location. By setting up

an initial position value when a user arrives, instead of a global searching the entire database of an area to get the initial position, we reduce much computation load.

Note that what we are actually making use of are the existence of features, or feature spatial distributions, to decide whether we are at some specific location or not; we do not have to recognize specific objects, e.g. doors, signs, etc., or utilize the geometric relationship among the objects in the environment in this 2D omnidirectional imaging based environmental modeling.

(2) 3D structure based environmental modeling

Real physical space is three dimensional; thus, it is natural to model the world with 3D sensors or via 3D reconstruction techniques, and then to use these 3D models to achieve reliable indoor localization. There are two major methods to construct 3D environmental models: Structure from Motion (SfM) and direct 3D sensing.

SfM is the process of estimating three-dimensional structure from two-dimensional image sequences [10]. Depending on the sources of the 2D images, SfM based localization approaches can be classified into two categories: crowd-sourced images based and user-captured images based localization. In indoor localization tasks for visually impaired people or the blind, the areas we want to localize are not popular places of interest for the general public. Therefore the images obtained from the Web are neither sufficient nor dense enough to model such 3D environments. So crowd-sourced image based 3D reconstruction is not appropriate for our purpose. In this thesis, we collect the images/video data by ourselves and use these visual information for 3D reconstruction.

The creation of the structure from motion work in this thesis involves three parts: GPU-based SIFT feature detection; pair-wise feature matching; and sparse as well as dense 3D reconstruction, which will be discussed in detail in Section 5.1.

The 3D environment information can also be obtained directly with portable and real-time 3D sensors, such as Google's Project Tango [8], Microsofts Kinect [9], ASUSs Xtion [10], Structure io [11], etc. We denote 3D environment modeling methods using above sensors as direct sensing method, differentiating it from structure from motion method.

One issue of the 3D data obtained by these devices is that their field of view of a single scan is limited. Sensors can only scan a very small local area in one moment, within a short range (0.8 to 4.0 meters) [12]. So we need to align these discrete and unregistered local 3D scans together for a complete environmental model.

A common way to deal with this problem is to combine and register all the local 3D data captured at different locations/times together and construct a global consistent 3D model under some assumptions, e.g. all global physical space is static and will not change during the localization and navigation procedure. More details will be discussed in Section 5.3.

1.2.2 Localization algorithms

Once the environmental models have been built, the newly captured input visual information can therefore be utilized for computing the location where these information are acquired. We denote the methods for aligning new visual input with global models as localization algorithms.

Depending on the modeling methods by which the environmental models are built, as discussed in previous subsection 1.2.1, the algorithms for localizing new input visual information are different correspondingly.

2D visual information based modeling uses image features as landmark representations, and the localization task essentially can be modeled as an information retrieval process [35]. 3D based modeling, however, relies geometric calculation for localizing new visual

[8] http://get.google.com/tango/
[9] https://developer.microsoft.com/en-us/windows/kinect
[10] https://www.asus.com/3D-Sensor/Xtion_PRO/
[11] http://structure.io
[12] http://get.google.com/tango/

information, for example, using 2D-to-3D matching or 3D-to-3D matching strategy. We will briefly introduce these two approaches in the following paragraphs, the details of which are explained in Chapter 4 and Chapter 5 respectively.

The 2D retrieval process includes the following steps: (1) capture one or more new images and extract representation features; (2) search features against the database of a scene; (3) find the most likely matched ones; and (4) finally output the localization results aligned with these features under certain matching criteria.

For the SfM models, given an input image, localization task is initialized by again extracting local features from the image and then finding which points in the model are matching with this image. After corresponding points are found, we can use the perspective N-points (pNp) algorithm [53] [34] to determine the camera location and orientation, i.e., the camera pose.

For direct dense sensing 3D models, for example, models captured by Google Project Tango Tablet device, a user can perform the localization task by capturing new RGB-D data at a new location via holding the Tablet, and registers the new data with the pre-built global 3D model using the ICP algorithm [108][32].

1.2.3 Assistive User interfaces

Assistive User interface is another important component in assistive technologies and solutions. Our localization systems and corresponding interfaces, even though can be used by general population, or some other special communities, such as elderly people, are designed mainly for visually impaired people.

There are several features which we shall take into consideration when designing assistive technology according to our corroboration with the visually impaired community. First, the interface shall be on the devices that are both light and portable. It's better if the solution can be embedded into their existing daily used devices, such as smart-phones. Visually impaired people already have a couple of things for carrying, e.g. cane, Braille printers, so light and

portable property is highly preferable. Second, the device shall be non-intrusive. No extra attention shall be drawn from the surrounding people, therefore the visually impaired can feel comfortable for a normal social life. Third, the design shall be low-cost. Many of the visually impaired people live with low-coming, or have already spent a certain amount of money in other assistive expenditures, so affordability is a critical feature.

Besides the overall system design, the assistive user interface is also an important component of our work. In this thesis, depending on the methods chosen, we have three devices and user interfaces: iPhone with customized iOS app, Google Glass with voice command control, and touch screen enabled Google Project Tango.

In the iOS app, since visually impaired people, especially blind people, cannot see the screen, and they usually use their smart-phones with the smallest lightness to save power, the key idea our design is to make it easy and simple to use. We lay a big bottom in the middle of the screen, and by pressing this bottom, the process of image capture, feature extracting, client-server communicating, matching and candidates aggregating will be executed automatically. Result will be read out to the users via voice feedback after localization result is returned. Also, since this method depends on that a user holds the smart-phone vertically in order to extract correct image features, we have a component for detecting the smart-phone's inertial measurement unit (IMU) data, and suggests the user to adjust the holding posture via voice if the holding tilt angle is larger than a threshold. More details about this will be discussed in Chapter 4.

In the Google Glass application, the user interface is divided into two parts: instructions input and resulting voice output. The localization service can be initialized by either voice command, or by tapping the side of the Glass. The result is read out to the Glass wearer via the bone conduction transducer. While in the Google Project Tango approach, we use similar design as the iOS app by creating a click-able area on the tango screen, and by tap this area, all the underlying processes are accomplished and results will be read out via voice feedback. More details are discussed in Chapter 5.

1.3 Overview

In the following Chapters, we will present our indoor localization research work from a very focused vision-based perspective, putting in the context of a broader spectrum of general vision and non-vision localization methods reviewed in the Related Work part. By utilizing the state-of-the-art portable devices and computer vision technologies, we try to investigate what machine vision methods can achieve for the assistive indoor localization task. The major contributions are summarized as follows.

C.1. **An Economic and Portable Solution**

- A novel combinations of a daily-used smart phone with a low-cost lens (GoPano) are used to provide an economic, portable, and robust indoor localization service for visually impaired people.

C.2. **A Compact and Distinguishable Omnidirectional Feature**

- New omnidirectional features (omni-features) extracted from 360 degrees field-of-view images are proposed to represent visual landmarks of indoor positions, and then used as on-line query keys when a user asks for localization services.

C.3. **Scalability for Storage and Computation via Cloud Computing**

- A scalable and light-weight computation and storage solution is implemented by transferring big database storage and computational heavy querying procedure to the cloud.

C.4. **GPU-accelerated Real-time Performance using Parallelized Algorithm**

- Real-time query performance of 14 fps is achieved with a Wi-Fi connection by identifying and implementing both data and task parallelism using many-core NVIDIA GPUs.

C.5. **Path-planning for Data Collection and Geometric Computing for Pose Refinement**

- Automatic path planning is developed for efficient environmental modeling by utilizing architecture AutoCAD floor plans, and a number of methods for refining localization results via 2D-to-3D and 3D-to-3D geometric matching.

The rest of the dissertation is organized as follows:

Chapter 2 presents a brief literature review on the state-of-the-art related work: (1) non-vision based indoor localization approaches, including GPS, Bluetooth, RFID, and WiFi-based indoor localization; (2) omnidirectional-vision-based indoor localization, within which omnidirectional imaging process, feature extraction, scene representation, image-retrieval based localization approaches are discussed; (3) other vision-based indoor localization, including Structure from Motion (SfM) based localization, direct 3D sensing based localization; and (4) assistive technology and user interfaces.

Chapter 3 describes the use of architecture drawings, especially AutoCAD architecture floor plans for automatically derive the traversability of paths within a building, which can be used as a guidance in modeling the whole environment. It consists of the following four steps: (1) the system reads an architectural floor plan (such as an AutoCAD file), extracts information of each entity (room, corridor and so on) and layers, and then stores them into a database; (2) an image-based analytics method is applied to extract each rooms layout-entity polygon; (3) the system identifies the geometric relations between neighborhood rooms and corridors, which allows a topological graph of the entire building to be computed; and (4) a 3D floor map and a traversable map are finally generated.

Chapter 4 introduces our framework of constructing a real-time assistive indoor localization system using omnidirectional GoPano lens and a smart phone. First, omnidirectional images obtained from device are preprocessed and distinguishable global features, Omni-Features, are extracted from these images for representing the scene landmarks. Second, the

omni-features are transformed, for the purpose of reducing storage requirement and computation time without losing localization accuracy, and indexed into environmental modeling database, which are aligned with their locations on floor plan. Third, parallelized multi-frame based querying methods are discussed for real-time finding the most similar candidates as well as aggregating a finalized location. Four sets of experiments are presented for demonstrating the accuracy and time performance of the solution.

Chapter 5 focuses on the visual information based refinement approaches in the indoor localization task from three perspectives. First, we discuss a structure from motion based indoor localization refinement method. Then we propose a multi-view omnidirectional geometry based localization refinement method. Finally, we introduce a direct sensing based localization refinement approach.

Chapter 6 summarizes the overall vision based indoor localization framework and some possible complementary refinement. Promising future directions are listed along the discussion of the trends in the hardware and software development of the visual information based localization system. Though this thesis uses a few devices to test the possibility of exploring different perspective of machine vision for solving the localization problem, in the near future, all these devices could be integrated into a single platform and provides a powerful hardware platform for all our current algorithms, as we can see from the recent released products in the industrial.

Chapter 2

Related Work

Indoor localization solutions, depending on what kinds of sensors are used for obtaining the environmental information, can be divided into two categories: non-vision and vision information based systems. As we discussed in Section 1.2, environmental sensing and modeling is the first step to begin a navigation task, which is then followed by processing and analyzing the obtained data with different kinds of computational methods. In this chapter, we first reviewed non-vision sensor based localization methods for practical assistive navigation systems. Then, we turn to the vision sensor based localization methods, especially the omnidirectional sensor and how it can be applied to localization and navigation tasks. Finally, we discuss the assistive technology in general and unique features about their user interface.

2.1 Non-vision-based indoor localization

In this section, we discuss the non-vision information based localization research work, especially for assistive localization purpose [46][13]. In general, according to the sensors used, localization can be divided into two major categories: GPS-signal-based methods and frequency-based methods. The state-of-the-art research on frequency-based indoor localization task using non-vision sensors include bluetooth-based indoor localization, RFID-based indoor localization and Wi-Fi based indoor localization [55].

(1) GPS based localization

The Global Positioning System (GPS), also known as Navstar GPS or simply Navstar, is a global navigation satellite system (GNSS) that provides geolocation and time information to a GPS receiver in all weather conditions, anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites [1] [2]. The accuracy of GPS in outdoor environments is around 2m [3], and the GPS signal in space will provide a worst case pseudo-range accuracy of 7.8 meters at a 95% confidence level, as shown in Fig.2.1. The actual accuracy a regular user can obtain actually depends on a lot of factors that are out of the providers control, including atmospheric effects, sky blockage, and receiver quality. Real-world data from the FAA shows that their high-quality GPS receivers provide better than 3 meter horizontal accuracy.

The localization tasks in outdoor environments are predominantly via GPS, such as widely used Google maps application [4]. However, this is not the case when it comes to the indoor environments. Due to the significant signal attenuation caused by construction materials, this satellite based GPS loses significant power indoors and therefore making such methods impractical.

(2) Bluetooth and RFID based localization

Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength UHF radio waves in the ISM band from 2.4 to 2.485 GHz) from fixed and mobile devices, and building personal area networks (PANs) [5]. Even though the bluetooth technology is built to transfer data, it is also capable of telling the distance between the transmitter and the receiver. Many companies are doing beacon based indoor navigation

[1] http://www.gps.gov/technical/ps/2008-SPS-performance-standard.pdf
[2] http://www.loc.gov/rr/scitech/mysteries/global.html
[3] http://www.gps.gov/systems/gps/performance/accuracy/
[4] https://www.google.com/maps/
[5] https://www.bluetooth.com

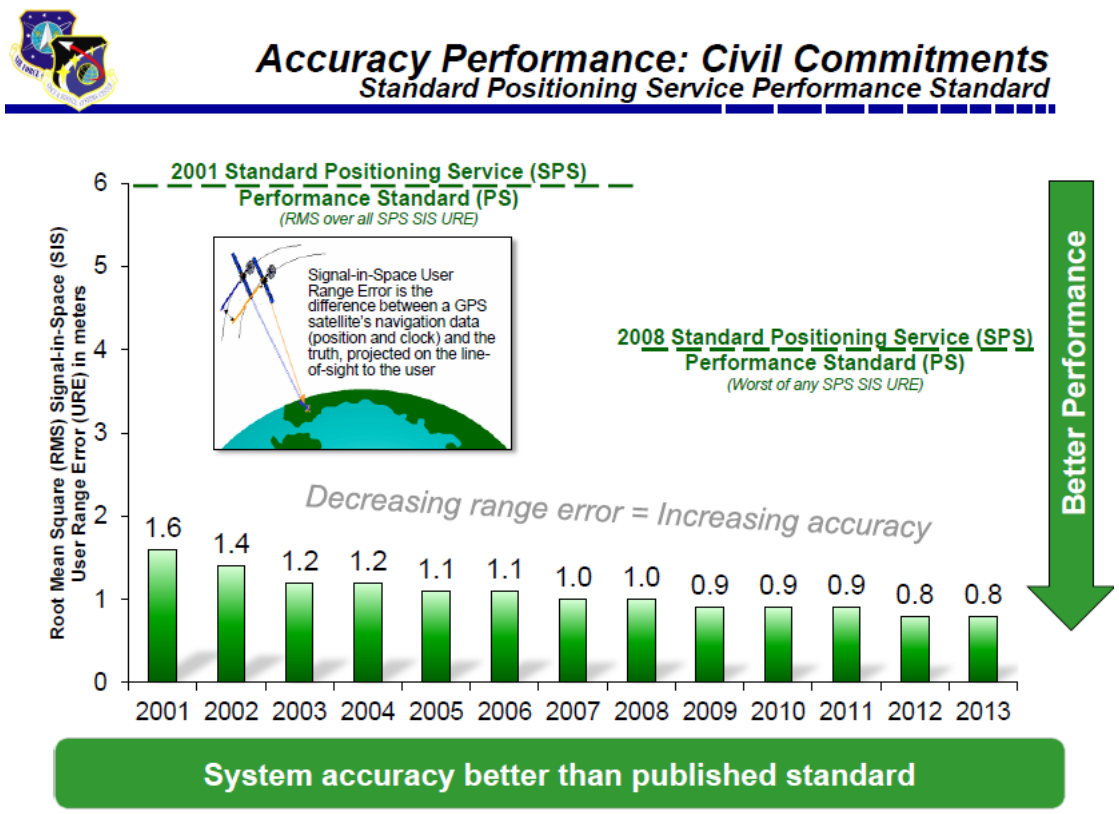


Figure 2.1: GPS accuracy in outdoor environment

systems. For example, SPREO, a US and Israeli company, founded in 2013 [6], provides a bluetooth beacon signal based indoor navigation service, and claims an accuracy of 1.5m.

The bluetooth indoor navigation system usually involves two parts: the hardware bluetooth beacons, and the mobile receivers, which are usually smart-phones, like iPhones or Android phones. The beacons are low cost, low energy chips and the battery can last for years (e.g. 5 years for Texas Instruments (TI) CC2540 Bluetooth System-on-Chip beacons). Fig. 2.2 shows an example of such a chip.

The smart-phone receives a signal from these beacons, and calculates the current location, the floor number, related maps, and possibly zoom in and zoom out functionality using triangulation. This information can be used for images/videos capture using the smart-phone for door and sign detection and localization [5][21].

[6] http://spreo.co/

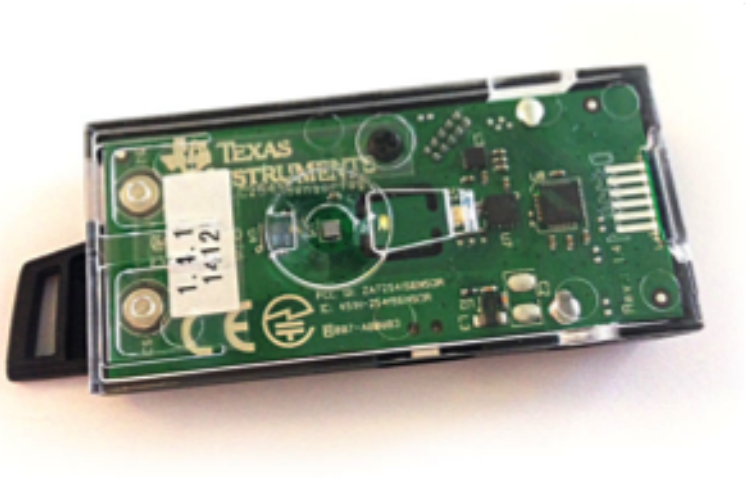


Figure 2.2: A sample bluetooth beacon chip

Radio frequency identification, or RFID, is a generic term for technologies that use radio waves to automatically identify people or objects. However, it can also be used for the indoor localization task [101][26]. The major idea is to use the electromagnetic field wirelessly to transfer data to automatically identify and track tags attached to objects. As opposed with bar-codes [61], RFID tags do not necessarily have to be within line of the sight of a RFID reader, and may be also embedded in a tracked object.

A classical RFID localization system uses labels or tags attached to the objects to be identified [12], and there is a bidirectional communication between a sender (also known as interrogator or reader), which sends signal to a tag and reads its response, and a tag. Tags can be passive, active or battery-assisted passive. The active tags have batteries built-in and can periodically transmit their signals. The battery assisted passive tags have a small battery inside but the battery is only activated when the tags are in the range of a reader. The passive tags are small and cheap, but they require a signal magnitude three times than the normal because they need to use the energy inside the signals to transmit the data back to the reader.

In Chumkamon [12], the RFID tags are embedded into stone blocks and put onto a footpath. By continuously detecting and tracking the RFID signal with a portable RFID reader, the app determines the current location by decoding the received signals and looking

it up in the database. Alternatively, each tag can be installed at signposts along a pathway, footpath or at the junction of the footpath. The RFID tags contain tag IDs and tag location information, and each location information is identified by a location area, a path, a link, or a node in terms of longitude or latitude.

Both the bluetooth and RFID localization methods need to modify the environment by attaching large amount of beacons or tags in order to function properly. When the localization service providing area expands, the cost would increase significantly, which limits the scalability of these methods. Meanwhile, changing the environment setting would arise other issues, such as aesthetic or legal issues. In addition, the accuracy of the localization service depends on the number of sensors mounted—for example, number of beacons or tags—this adds to the burden of installation as well as maintenance of these systems.

(3) WiFi based localization

WiFi (also, Wi-Fi or Wifi) is a local area technology that allows an electronic device to exchange data or connect to the internet using specific radio frequencies, e.g. 2.4GHz UHF or 5GHz SHF. Besides its data exchange function, similar with bluetooth, WiFi is also used by many to localize or navigate devices [8][50][6].

For example, Google uses WiFi access points' geolocation and MAC addresses to improve its location-aware service accuracy. The locations of the access points are recorded by a Street View car when WiFi signals are received passively from these points. Databases are created in Google's data center, and when a new user requests a location service, the user's nearby AP's SSIDs and MACs will be sent to Google, and Google searches its databases to return location data to the user [7].

The detailed steps are shown in the following: Step 1: The user's device sends a location request to Google's location server, and all the nearby MAC addresses that are visible to the device are included. Step 2: The location server compares the received MAC addresses and searches its databases to identify associated geocoded locations, for example providing

[7] https://goo.gl/J7BmTn

the latitude and longitude. Step 3: The location server then uses the geocoded location to triangulate the approximate location of the user. Step 4: The approximate location is geocoded and sent back to the user's device.

As another example, students at Stanford University created a company—WiFiSLAM—with a similar patented WiFi based technology to do indoor navigation [8].

The advantage of WiFi based technology is that it does not require the environment to be installed with extra devices, and it is supported by almost all smart-phones and wearable devices. However, to successfully localize a device, at least three known WiFi access points are required, and to achieve higher accuracy, having more devices are preferred. This requirement is not always satisfiable in all indoor environments. In addition, to construct a robust localization system, the access point's database needs to be updated periodically, and if some access points are added or deleted from the environment, it also needs to notify the database server in time.

Though the above methods to some degree can obtain information for localization and navigation, few can match the contextual information provided by 2D/3D visual sensors. Sensor technologies can help, but equipping a visually impaired user with machine vision capabilities comparable to human vision is always best [79]. In the rest of this chapter, we will discuss the 2D and 3D visual information based indoor localization approaches.

2.2 Omnidirectional-vision-based indoor localization

It is more natural for human beings to utilize vision, compared with using other modalities, to perceive the environment and carry out daily tasks. Visual information based localization is also generating more and more interest in both the computer vision community and the robotics community to achieve real-time and accurate localization [92][88].

Mobile device cameras and consumer level 3D sensors are becoming very cheap and widely used in our daily lives. A majority of the existing vision based localization services use 2D

[8] https://angel.co/wifislam

images or 3D RGB-D data from these devices to perform the task. However, a regular camera has a narrow field of view (FOV), and this limits its capacity to perceive as much information as possible at a given time. In particular, the narrow FOV makes the aiming of the camera a serious problem for blind users. Sufficient visual information plays a key role in achieving an accurate, fast, and robust localization goal, so this leads us to review how the wide FOV camera systems, especially omnidirectional vision systems can do to improve the performance of localization task.

A human being's FOV is about 120 degrees horizontally, and around 135 degrees vertically. The field of view of a typical smart-phone is only 50 degrees horizontally and 40 degrees vertically. If images are captured with different viewing angles around the same location, the scenes covered by the images will differ a lot. Meanwhile solving the matching problem of multiple images with motion parallax would be much easier if the coverage of the images is similar rather than between images with dramatic different coverage. This is why we utilize omnidirectional imaging system.

Omnidirectional imaging is usually achieved by integrating a lens and a mirror, or catadioptric optics, and a perspective camera [24][70]. According to whether there is a central focus or not, it can be divided into two categories: central imaging system, as shown in Fig. 2.3 and non-central imaging system Fig. 2.4 [85]. There are already a number of portable omnidirectional imaging systems available on smart phones; a typical configuration is to mount a special mirror on top of a smart-phone camera [9], whose cost is under a hundred dollars.

In general, an omnidirectional-vision-based localization system usually includes three modules: image capture with omnidirectional lens/mirrors, environment mapping or model construction, and feature extracting and searching. Though different researchers may use variant combinations of methods for each module, almost every system has these three components.

[9] http://www.gopano.com/products/gopano-micro-iphone5

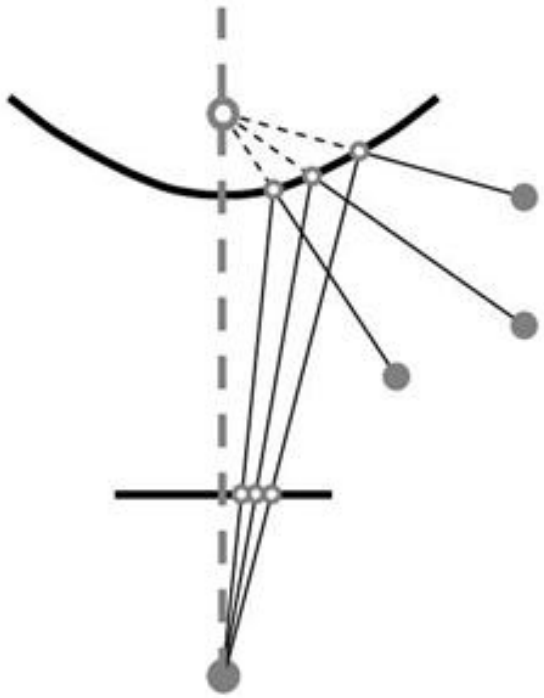


Figure 2.3: Central imaging system

Omnidirectional imaging is the process of using one or multiple omnidirectional imaging systems to move along the area where we want to provide localization services, and capture as well as store visual information for the purpose of model construction and/or real time searching [110][71]. Since the system does not estimate location for only one static spot, it should be mobile and have the ability to provide continuous localization services. The most widely used omnidirectional imaging systems nowadays are wearable devices or smartphones with specially designed omnidirectional lens or mirrors built into the original system or mounted as accessories. To provide stereo information, two or more omnidirectional imaging systems may need to cooperate and output the 3D environment information [17]. The sensing procedure can be carried out by robots, developers, or even users depending on the methods used in the environment modeling process.

Environment mapping or model building is the process of constructing a one-to-one or many-to-one relationships between the 3D points in the real physical space and points in the digital space [11][36][35]. If a 2D model is applied, for example, using a floor plan, we can

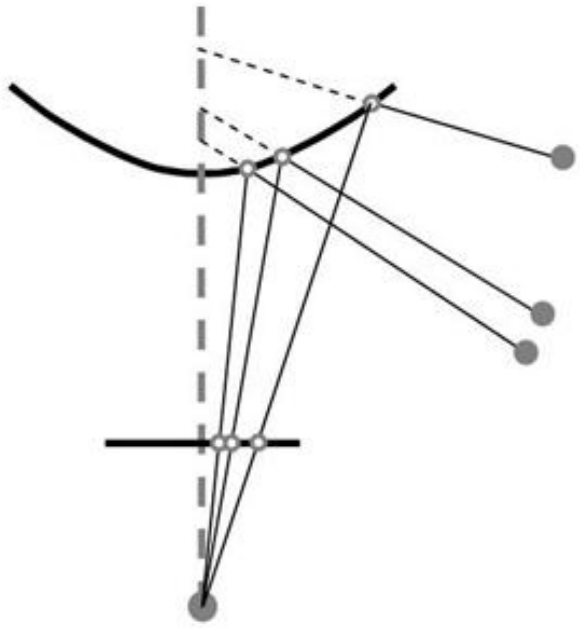


Figure 2.4: Non-central imaging system

project the 3D space into a 2D plane, and all the 3D points at the same location but different height will project to the same 2D point [34]. For the purpose of localization, in most cases, a 2D map is sufficient and can satisfy all the localization service requirements. However, in some cases, for example, multi-floor buildings, a single map is not enough. Though this can be solved by providing one map for each floor, many discrete 2D virtual spaces are involved, and the positions between the floors, for example, the stairs between floors, cannot be distinguished. To solve this problem, some researchers use 3D digital models and all the 3D points in the real physical space have a one-to-one corresponding point in the digital space [32][52].

Feature extraction and searching is the process of extracting representations of local images/3D points and using them to construct the digital space model or to find the correspondence in the pre-built digital space. Even though images, videos or 3D points generated by computing devices themselves are already discrete representations of the 3D physical space, they are still, in most cases, not concise and descriptive enough for efficient matching or searching. We need to further extract features for these images, videos or sparse

3D points. 2D image feature extraction and matching have been studied for decades in the computer vision community and are relatively mature subprocesses in visual information based localization [3][83]. 3D features matching and searching or 2D/3D combined feature matching and searching are relatively new and have attracted a lot of attention in recent years [84][82].

Appearance-based localization and navigation have been studied extensively in the computer vision and robotics communities using a large variety of methods and camera systems. Outdoor localization in urban environments with panoramic images captured by a multi-camera system (with five side-view cameras mounted on the top of a vehicle) is proposed by Murillo *et al* [68]. Another appearance approach proposed by Cummins and Newman [15] is based on Simultaneous Localization and Mapping (SLAM) for a large scale road database, which is obtained by a car-mounted sensor array as well as a GPS. Kanade *et al* [48] [4] localize a vehicle by first driving through a route and then comparing a captured image in the current drive with the images in the database created from the first drive. These systems deal with outdoor environment localization with complex camera systems. In our work, we focus on the indoor environment with simple but effective mobile sensing devices (smart phone + lens, or Google Glass + SfM model) to serve the visually-impaired community.

Since a single image from normal camera has very limited field of view, and thus cannot make use of the entire visual information available, creating mosaic or panorama images from discrete images are studied by many researchers [30][65]. A visual nouns based orientation and navigation approach for blind people was proposed by Molina *et al* [64], which aligns images captured by a regular camera into panoramas, and extracts three kinds of visual nouns features (signage, visual text, and visual-icons) to provide location and orientation instructions to visually impaired people, using visual noun matching and PnP localization methods [53]. In their work, obtaining panoramas from images requires several capture actions and relatively heavy computation resources. Meanwhile, sign detection and text recognition procedures face a number of technical challenges in a real environment, such as

illumination changes, perspective distortion, and poor image resolutions. In our research work, an omnidirectional lens GoPano [25] is used to capture panorama images in real-time, and only one snapshot is needed to capture the entire surroundings rather than multiple captures. No extra image alignment process is required, and no sign detection or recognition is needed.

Another related navigation method in indoor environments is proposed by Aly and Bouguet [2] as part of the Google Street View service, which uses six photos captured by professionals to construct an omnidirectional image at each viewpoint inside a room, and then estimates the camera pose and moving parameters between successive viewpoints. Since their inputs are unordered images, they construct minimal spanning tree among the complete graph of every viewpoint to select triples for parameter estimations, which is computational intensive. In our method, since we use sequential video frames, we do not need to find such spatial relationships between images, therefore the computation cost is reduced for a real-time solution.

Representing a scene with extracted features for different purposes, such as image classification or place recognition, has been studied by many researchers [19, 54, 73, 102]. An early work in representing and compressing omnidirectional images into compact rotation invariant features was proposed by Zhu *et al* [111], where the Fourier transform of the radial principle components of each omnidirectional image is used to represent the image. Different from [111], based on the observation that an indoor environment usually includes a great number of vertical line segments, we embed these vertical line segment distribution information into a one-dimensional omnidirectional feature, and then use the Fourier transform components of these features as the representation of omnidirectional images. Another major difference is that we aim to find a user's location and orientation from each omnidirectional image, whereas in [111], only six road types are classified using a neural network based approach.

Another similar research area is image retrieval. Direct image retrieval entails retrieving those image(s) in a collection that satisfy a users need, either using a keyword associated

with the image, which forms the category of concept-based image retrieval, or using the contents (e.g. color, shape) of the image, which forms the category of content based image retrieval [44]. The applications of content based image retrieval include sketch-based image searching [39], changing environment place recognition [63], etc., but very few research are conducted for using the content based image retrieval for localization purpose, especially indoor localization. One major reason is that to represent a specific location using images, all the visual information around this location have to be collected and stored to make this landmark distinguishable, which requires many images if using a normal Field Of View (FOV) camera, instead of a single image. Also, the features used for retrieving from the input image should depend on the location only, independent of the cameras orientation, but this is hard for normal FOV cameras, because the scene images are usually captured from different perspectives, and they therefore generate different features even though images are captured at the same location.

2.3 Other vision-based indoor localization

Utilizing vision information for the purpose of localization is attracting more and more attention these years in both industrial and academic community [23]. Structure from Motion (SfM), a technique to create street 3D models in the outdoor environment, and recognize the places utilizing images from Internet is one of them and is studied by a number of researchers [93][105][81].

We denote the points in the 2D images as features, and a point in the 3D points in the SfM model as points, which may be associated with many features. A general localization problem can be stated as: given an input image, find which points in the model are matched with this image. If corresponding points are found, we can then determine the camera location and orientation, i.e., the camera pose [34].

Since the cameras taking the images are not calibrated, we cannot determine the final scale of the 3D model built by SfM from images alone. However, it is still meaningful to localize input images within these 3D models up to a scale. To solve the scale problem, we need extra knowledge about the environment. Kume resolves the scale problem by using odometry [47]. Other methods [106][34] to solve this problem is to relate a 2D image feature with the 3D physical world by manually labeling it, for example, manually relate the SIFT feature with a 3D point. The scale of the model therefore can be determined by using known 3D points.

Lee *et al* [52] [51] use Microsoft Kinect to construct a 3D voxel model for an indoor environment and analyze whether each voxel is free or occupied. Based on this information, a decision is made and then the system notifies the user what should be the next move. This system, however, requires an extra non-daily used device Kinect, which is not easy to integrate into blind people's daily life.

Other researchers use Bag of Words (BoW) [9] or ConvNet features [89] to represent outdoor environments for localization. However, few researchers focus on the indoor scenarios, especially for assistive localization purpose.

2.4 Assistive technology and user interface

Assistive technology (AT) is an umbrella term that includes assistive, adaptive, and rehabilitative devices or systems for people with disabilities and also includes the procedures used in selecting, locating, and utilizing them. Assistive technology promotes greater independence by enabling people to perform tasks that they were previously unable to accomplish, or had great difficulty accomplishing, by providing enhancements to, or changing methods of interacting with, the technology required to accomplish such tasks. Many factors are important for a successful assistive technology–consideration of user opinion in selection, easy device procurement, device performance, and change in user needs or priorities [77].

In this thesis, we do not intend to review all the devices/systems used for people with disabilities in general. Instead, we focus only the visually impaired people (VIP) and the localization task for VIPs in their daily life.

The user interface (UI) is the space where interactions between humans and machines occur for certain functions. The purpose of this interaction, especially in the assistive technology field, is to allow effective operation and control of the machine from the human end, and at the same time allow the machine simultaneously to provide feedback information that aids the operators' decision-making process.

Mobile and wearable devices are cheap and ubiquitous nowadays, which accelerate the advancement of both general computer vision research and assistive applications. For example, Farinella *et al* [20] use Android phones to implement an image classification system with DCT-GIST based scene context classifier. Altwaijry *et al* [1] apply Google Glass and develop an outdoor university campus tour guide application system by training and recognizing the images captured by Glass camera. Paisios [75] (a blind researcher) creates a smart phone HCI for the Wi-Fi based blind navigation system. Manduchi [59] proposes a sign-based way-finding system and tests the blind volunteers with smart phones to find and decode the information embedded in the color marks pasted on the indoor walls. All these research work use mobile or wearable devices and design either touch screen or voice command (feedback) based interface for communicating with the visually impaired people. However, there is very few research work on designing user-friendly smart phone apps (e.g. iOS app) for helping visually impaired people to localize themselves and navigate through an indoor environment.

Chapter 3

Path Planning for Data Collection

In order to localize a user in an indoor environment, the first question we shall ask is: how can we effectively and efficiently select paths to model an environment using our omnidirectional imaging system? We need a path planning algorithm to determine paths for capturing omnidirectional images in order to model an environment.

This path planning work, even though intended for convenient environmental modeling, is also valuable for visually impaired people themselves. When a visually impaired person enters an indoor environment for the first time, for example, a multi-floor building, he/she needs knowledge about the structure of the building or layouts of each floor to successfully navigate within it for reaching the desired destination, such as a room. This knowledge can be acquired by multiple ways: self-exploring for multiple times and memorizing it; asking other people within the building; asking other people who are not within the building but remotely connected, for example via video [74], or independently learning the architecture information before leaving. The first three methods are effective sometimes, but at the cost of losing users' independence, which is one of the core spirits need to consider when designing an assistive technology. Learning the architecture maps of a building before physically entering the building, i.e. pre-journey, is an effective way for them to navigate inside the building or using existing localization and navigation services [76][98][38].

In this Chapter, we are utilizing architecture drawing of buildings for achieving automatic path planning for both pre-journey planning by visually impaired people and environment modeling by developers [90][91]. Architecture drawing is a technical drawing of a building, usually using Computer Aided Design (CAD) software (e.g. AutoCAD) and is available for buildings built in the past three decades. They are widely used by architecture and others for a number of purposes: to develop a design idea into a coherent proposal, to communicate ideas and concepts, to convince the clients the merit of a design, or as a record of a completed construction work.

Using the same path planning algorithm for both environment modeling and pre-journey planning also makes the modeling and localization stages more consistent. Once the traversability between any two places, e.g., the entrance and a restroom, is calculated, single-path or multi-path based omnidirectional imaging modeling, can be performed for providing robust indoor localization service; details in modeling will be discussed in 4.3. Then in the localization stage, when the user follows a path very similar to the path that has been planned for modeling, the image-based localization would be much more effective.

The system of traversalbility calculation includes the following four steps. (1) The system reads an architectural floor plan (such as an AutoCAD file), extracts information of each entity (room, corridor and so on) and layers, and then stores them into a database. (2) An image-based analytics method is applied to extract each room's layoutentity polygon. (3) The system identifies the geometric relations between neighborhood rooms and corridors, which allows a topological graph of the entire building to be computed. (4) A 3D floor map and a traversable map are finally generated. The system diagram is shown in Fig. 3.1. In the following sections, we are going to discuss these four steps in details.

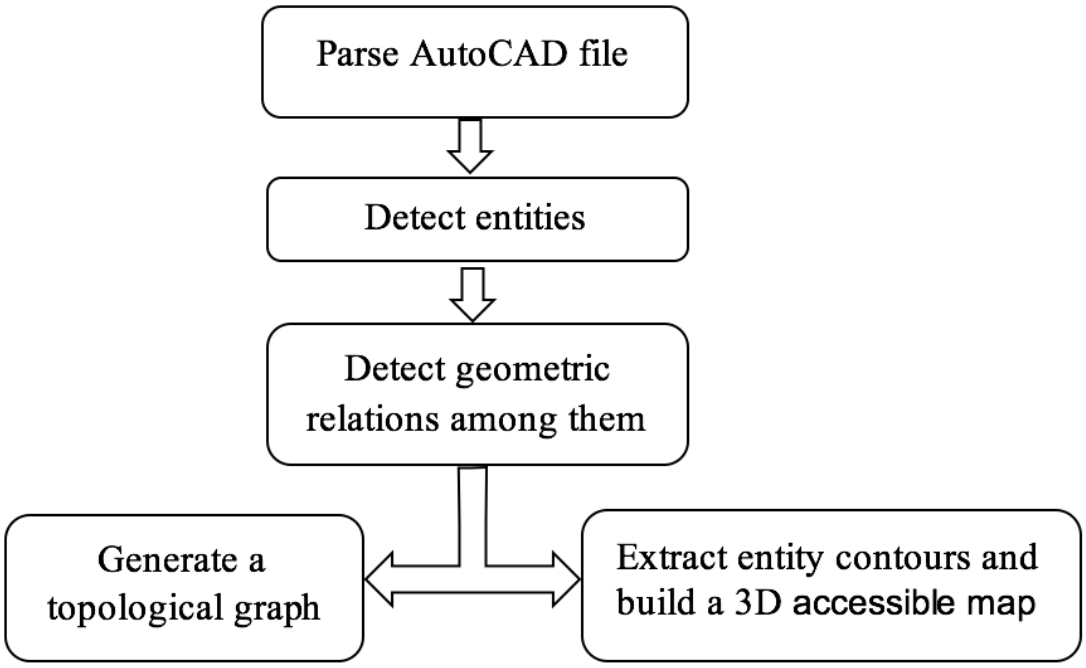


Figure 3.1: Automatic path planning system diagram

3.1 Architecture information extraction

Architecture drawings are made according to a set of conventions, which include: particular views, annotations, cross references, units of measurements and scales, and so on. Fig. 3.2 shows an example of an AutoCAD architecture floor plan.

In this work, mainly the architecture floor plans are used. Though the floor plan in AutoCAD is a vector image (e.g. lines, arcs and other simple geometric shapes), the information of the entities' boundaries is not available. So the correspondences between the geometric shapes (in the forms of polygonal boundaries) and the entities are unknown when the architectural floor map is drawn. Therefore, we first need to extract entity polygons from the AutoCAD file and then analyze the geometric relations among different entity polygons in the form of a rendered floor-plan image in order to build the topological map of the entities.

Figure 3.2: An architecture floor plan example

The architecture information are not stored in the format of images, but in a standard format called Drawing Interchange Format (DXF) format. We first have to extract layers from a DXF file, based on the DXF specifications [1]. Different layers store different perspectives of the building information. For example, one of the layers is text annotations that includes the room numbers, and another layer could provide the locations of room centers in the floor map.

In addition, we extract semantic information (name tags of the rooms, corridors, etc.) from the AutoCAD file, which are the locations of entities, including rooms, exits, elevators, restrooms and some landmarks. The location information are important since they can

[1] http://www.autodesk.com/techpubs/autocad/acadr14/dxf/dxf_reference.htm

provide the center location of the entities, which could be used as the seeds in the region growing process.

After all the above information are extracted, we store them into a database for an indoor environment, for example, one single floor. Note that some layers are not related and therefore not needed in our path planning task, hence we do not need to spend time and effort to process them. For example, electricity map describes how the electricity wires go through the building, which is irrelevant and therefore removed in this step.

The layers we are interested the most is the floor map which shows the exact geometry relationship of the corridors and the rooms up to a known scale. Instead of manually walking through a building in person and finding the paths needed for modeling, using the floor map provides a much flexible way of designing the desired paths without spending time into complex buildings and taking notes for traverse maps. This is also more efficient when the area of modeling scales up to multiple buildings or multiple university campuses.

3.2 Layout and entity detection

Note that the above layout and entity information are stored in the format of text, not as visual format such as images. We therefore provide a rendered map by generating our version of floor plan image using previous generated database data. One sample of our rendered floor plan image is shown in Fig. 3.3. In this floor plan version, only useful information are drawn, and all the other unnecessary information are not rendered. Also note that the rendered image is not just for visualization, it is mainly used for analyzing the geometric and topological relations of the entities.

For further relating the entities' locations and area boundaries, we create a polygon for each entity using a region growing algorithm [62][80]. Region growing is a region-based image segmentation method. It first selects a set of seed points, and then by examining the pixel neighbors of the seed to determine whether they shall be added into the region. The major

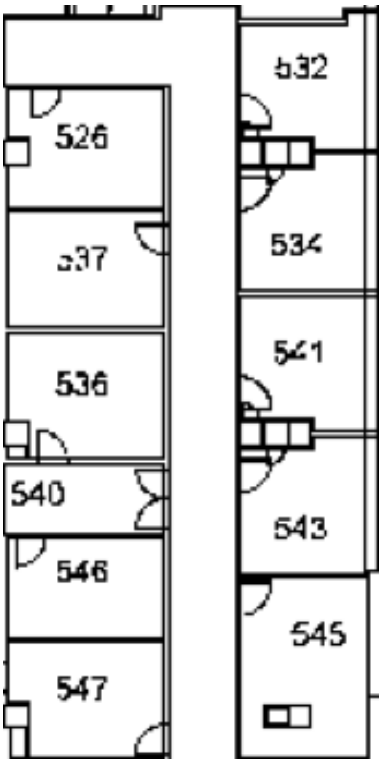


Figure 3.3: A sample floor plan visualization result

goal of this algorithm is to partition an input image into regions, which is formalized as follows:

(a) $\cup_{i=1}^{n} R_i = R$;

(b) R_i is a connected region, $i = 1, 2, ..., n$

(c) $R_i \cap R_j = \emptyset$ for all $i = 1, 2, ..., n, i \neq j$

(d) $Property(R_i) = true, i = 1, 2, ..., n$

(e) $Property(R_i \cup R_j) = false$ for any adjacent region R_i and R_j

In other words, (a) means region growing can be treated as a classification problem, and every pixel shall fall into a region. (b) constraints that all the pixels in a region shall be connected in a predefined way. (c) requires that all the region shall be disjoint with each other. (d) and (e) mean that pixels in the same region shall share the same property, e.g. the same grey value range, and the pixels in different regions shall have different property.

To employ the region growing algorithm, first the floor map image is converted into a binary image with wall partitions drawn in 0s (black), and open space, such as rooms and corridors are 1s (white), as shown in Fig. 3.3. Then we query semantic information obtained in the first step to obtain the entity number and the centroid of an entity. Starting with the centroid, we apply a region growing implementation [80], to scan through the empty space in the entity and hence the image region of each entity is obtained. The process is repeated until all known entities (in the database) have been processed and the labeled entry are saved into the entity list.

Because some entities may not be available in the database, for example, corridors may not be identified in the database, they are still empty space in the image. Then we can scan each pixel in the floor image. For any empty pixel, we run the same region growing method. We store the labeled entity into the entity list if its area is greater than a threshold (> 50 sq. feet). The above process is repeated until there are no more empty area in the floor image. Fig. 3.4 shows the result after region growing is applied.

3.3 Entity geometric relations calculating

In the previous sections, the AutoCAD files are parsed, visualized and segmented using region growing algorithm. In this section, we are going to calculate the geometric relationships between the segmented entities (rooms, corridors, etc.) by connecting the labeled entities with neighborhood entities and build a topological graph.

We assume that entities are connected with doors, and each door will connect only two neighbored entities. First the semantic information for the positions of doors are queried, and then we start to process and extract relationships between rooms and corridors. Note, the corridors are usually not marked in the AutoCAD files so we declare any open space connecting multiple rooms as a corridor.

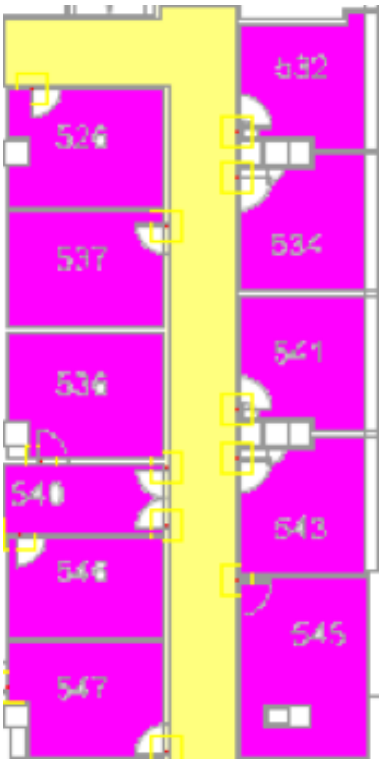
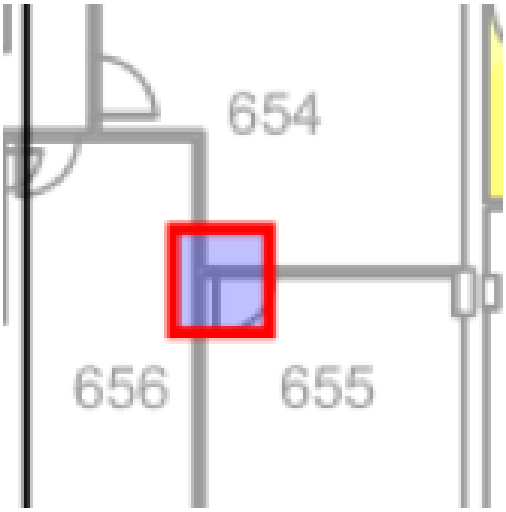


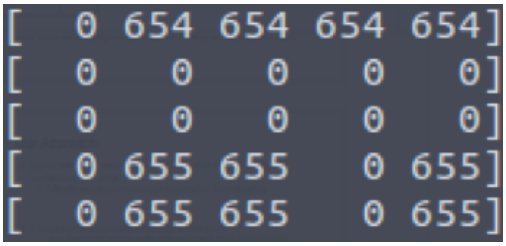
Figure 3.4: A sample region growing result

The door information plays an important role in identifying the geometric relations between two entities, such as two neighborhood rooms. Here is an example of how the door's information is utilized for calculating entity geometric relationship. For each door position $d(x, y)$ (such as the one shown in Fig. 3.5a), we create a Region Of Interest (ROI) $(x - p, y - p, x + p, y + p)$ where p is the padding initialized to 1. If there is only 1 unique nonzero pixel, then we increase p by 1 and create a larger ROI. This process is repeated until the ROI contains at least 2 unique nonzero pixels (a door connects two entities, represented by the two unique nonzero pixels).

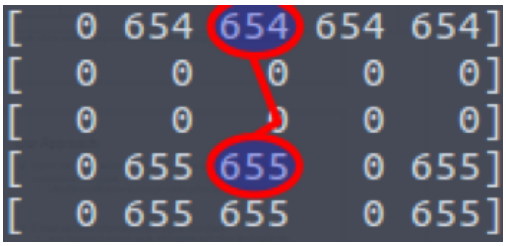
In order to make sure that there are only two entities connected through the door, two entities with the smallest distance from the ROIs center (the door) are identified. Fig. 3.5c shows an example. In this local area, there is a door connecting room 654 and room 655, as shown in the Fig. 3.5a. We increase the size of the ROI until we find two unique pixel values, i.e. 654 and 655, and assert that room 654 and 655 are connected via the door. Note



(a)



(b)



(c)

Figure 3.5: (a) A small section of a map. The red box shows the ROI around a door of the room 655. (b) The pixel values within the ROI. (c) The two unique room numbers nearest to the ROIs center are identified, and room 654 and 655 are connected in the topological graph

that if we increase the size of ROI, room 656 will be included into the region, but since we assume that a door only connects two rooms, we therefore only pick up the two numbers (654 and 655), which have the smallest distance to the ROI center.

The above process is repeated on all doors in the database, and the initial geometric relations among entities are built. A topological graph is therefore constructed, with each entity as a vertex and each door connecting with two entities as an edge. Fig. 3.6 shows

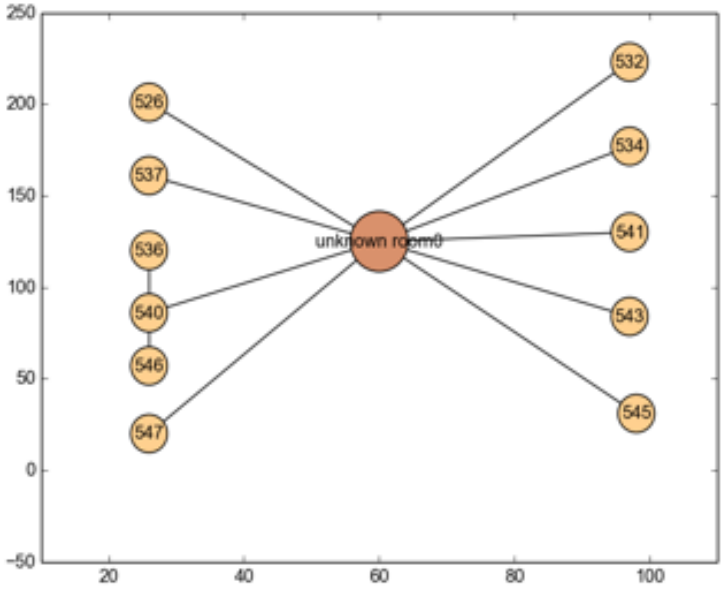


Figure 3.6: Entity geometric relationship for a sample input AutoCAD map

the graph generated from Fig. 3.5. The traversability problem can be therefore modeled as searching the shortest path in the graph and can be calculated using the Dijkstra's algorithm [16].

3.4 Contour extracting and 3D accessible map building

After we perform the region growing method and find the shape of each entity, we perform contour extraction using OpenCV [2], which computes the contours of each entity polygon. Each contour is simply a list of vertices of the polygon. We iterate over each vertex and insert them into a file using JSON format [3]. The content of the JSON file is structured in the following way, for each entity.

Entity n: {*the coordinates of polygon vertices, the coordinates of door(s)*}

[2] http://opencv.org
[3] http://json.org/example.html

The JSON file is loaded into the Unity3D engine and a 3D floor map is constructed automatically. The traversable map is built and an optimal route can be computed by using A^* algorithm [27].

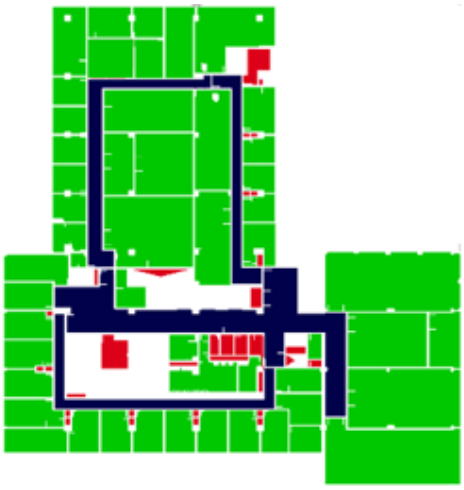
We have performed the proposed algorithm on an AutoCAD file of a complex campus building as another example. We still use the AutoCAD map as shown in Fig. 3.2. The previous four steps are carried out one-by-one with this map: we first parse the AutoCAD file, and then extract useful layer and semantic information, which are stored into a database. Afterward, we render a new floor image that only includes wall structures from the database. The region growing method is applied into the new floor image and entities are identified from the image. In Fig. 3.7a, the green regions represent rooms and blue regions represent corridors. Once the entities are extracted, the geometric relations among entities are calculated and a topological map is successfully built. Furthermore, the contour of entity polygons in the floor image is extracted and saved into a JSON file. A 3D traversable floor map is built in the Unity3D[4] environment, as shown in Fig. 3.7b, and a turn-by-turn navigation direction can be calculated within Unity3D.

We manually verify the accuracy of the topological and the traversable maps. For the topological map, we compare it with the original AutoCAD map. We derive a topological map from the AutoCAD map and compare it with the one generated by our proposed algorithm, and they match correctly. For the 3D traversable map, we project it into a 2D space and it aligns with the original AutoCAD map correctly as expected.

To test the performance of the topological and the 3D traversable maps, each time we randomly select two entities from the AutoCAD map and we manually calculate the shortest path between two entities. We then compute the navigation summary, using the Dijkstra's algorithm [16] from the topological map, and the turn-by-turn directions, using A^* algorithm [27] from the 3D traversable map. We compare the three paths and they also match correctly.

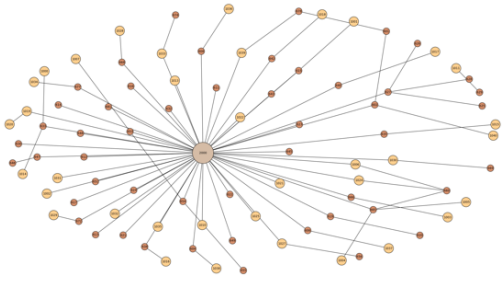
[4] https://unity3d.com

As a summary, given any building available with AutoCAD documentations, we then can detect the traversability between any two entities A and B (e.g. offices and restrooms). Also, we can generate turn-by-turn instructions about how to navigate throw the building from A to B, even though the person—for either a visually impaired person or environment modeling staff— who has never been to this building before. Therefore the proposed system can enable our administrative staff to utilize the omnidirectional imaging system and model the indoor environment, and our visually impaired users to navigate the same environment, more efficiently and effectively.



(a)

(b)



(c)

Figure 3.7: (a) A visualization of the lookup table after region growing is performed on the input AutoCAD map in Fig. 3.2. (b) The 3D traversable map rendered in Unity3D. (c) The topological map, each node is represented by a note and an edge represent the connectivity between two entities

Chapter 4

Omnidirectional-Vision-Based Indoor Localization

In this chapter, the work of constructing a real time assistive indoor localization system using omnidirectional GoPano lens [1] and a smart phone (iPhone) [2] is introduced and detailed experiment results are provided to demonstrate the accuracy and robustness of the solution.

The system consists of a mobile vision front end with a portable panoramic lens mounted on the smart phone, and a remote image feature-based database of the scene on a GPU-enabled server. Compact and effective omnidirectional image features are extracted and represented in the smart phone front end, and then transmitted to the server in the cloud. These features of a short video clip are used to search the database of the indoor environment via image-based indexing to find the location of the current view within the database, which is associated with floor plans of the environment. A median-filter-based multi-frame aggregation strategy is used for single path modeling, and a 2D multi-frame aggregation strategy based on the candidates distribution densities is used for multi-paths environmental modeling to provide a final location estimation. To deal with the high computational cost in searching a large database for a realistic application, GPUs are usually involved in

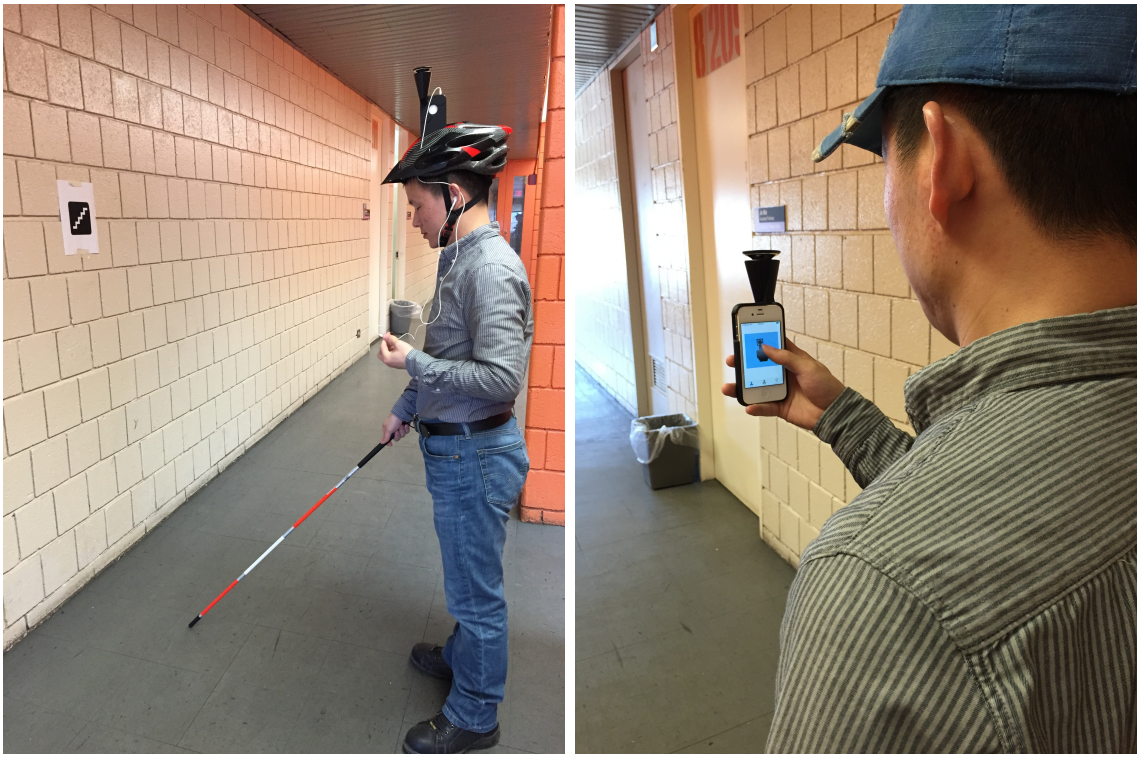
[1] http://eyesee360.com

[2] https://support.apple.com/kb/sp587?locale=en_US

Figure 4.1: Omnidirectional-vision-based indoor localization system diagram

an implementation [104, 107]. In our work, data parallelism and task parallelism properties are identified in the database indexing process, and computation is also accelerated by using multi-core CPUs and GPUs. User-friendly HCI particularly for the visually impaired is designed and implemented on an iPhone, which also supports system configurations and scene modeling for new environments. Experiments on a database of an eight-floor building are carried out to demonstrate the capacity of the proposed system, with real-time response (14fps) and robust localization results.

A system overview is discussed before we move into each Section for details. The omnidirectional assistive localization system uses a client-server architecture, as shown in Fig. 4.1. The server stores the environmental databases, and has an indexing program running all the time, waiting for the queries from the client side. The client side is implemented on a smart phone as an app, such as an iOS app. The client side hardware consists of a

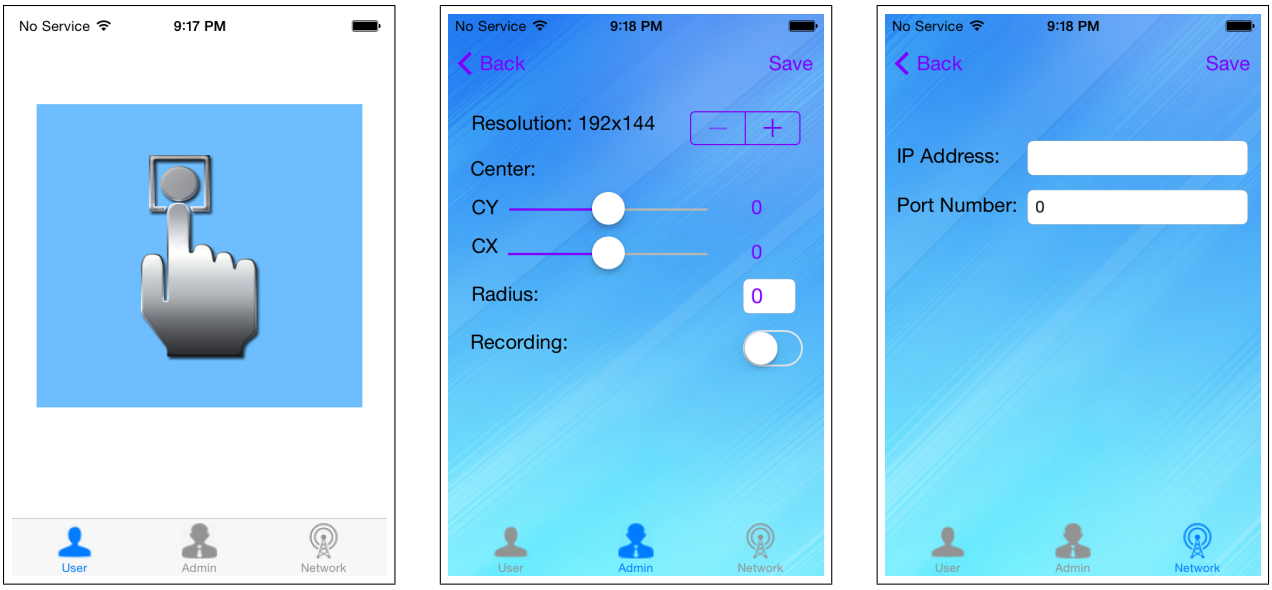


(a) (b)

Figure 4.2: Two system working modes

smart phone and an omnidirectional lens, which is mounted on the phone with a case. In our implementation, we use an iPhone and a GoPano lens.

There are two possible ways of using this system. One is hands-free mode, as shown in Fig.4.2(a). The camera is mounted on the top of a bicycle helmet worn by a blind user. The other is a hand-held mode, as shown in Fig.4.2(b). Localization query action is completed by a single pressing on a big button in the middle of the smart phone's screen, as shown in Fig. 4.3, where Fig. 4.3a is the major interface for the users, and Fig. 4.3b and Fig. 4.3c are interfaces for the administrative staff for adjusting configurations. During a usability study, we received a lot of feedback from the blind community for the project. Originally, we would like to mount the smart phone onto a helmet, so users do not need to hold phones by themselves. However, according to their feedback, the majority of blind people prefer to hold the phone by hand to avoid unnecessary attentions. Also, since the users only need to hold the smart phone for a few seconds when a localization service is needed, they do not



(a) (b) (c)

Figure 4.3: iOS app interfaces

have to hold the phone all the way while walking. As a result, in this thesis, we will mainly focus on the second working mode.

The building of the system includes two stages: the modeling stage and the querying stage, as shown in Fig. 4.4. In the modeling stage, the system developer carries the smart phone and moves along the corridors with an even normal walking pace (0.5m/s to 2m/s), covering and recording the video into a database. Geo-tags (e.g. physical locations of current frames) are manually labeled for associated key frames. Motion estimation algorithms can be used in the future to ease the constraint of linear motion. In the querying stage, a visually impaired user can walk into the area covered in the above modeling stage and take a short video clip. The smart phone extracts video features and sends them to the server via a wireless connection. The server receives the query, searches the image candidates in the database concurrently, and returns the localization and orientation result to the user.

For the modeling stage, we use all the frames of the video to compose the database in order to make full use of the visual information and obtain the highest possible localization resolution for each scene. During the labeling procedure, we select associated key frames,

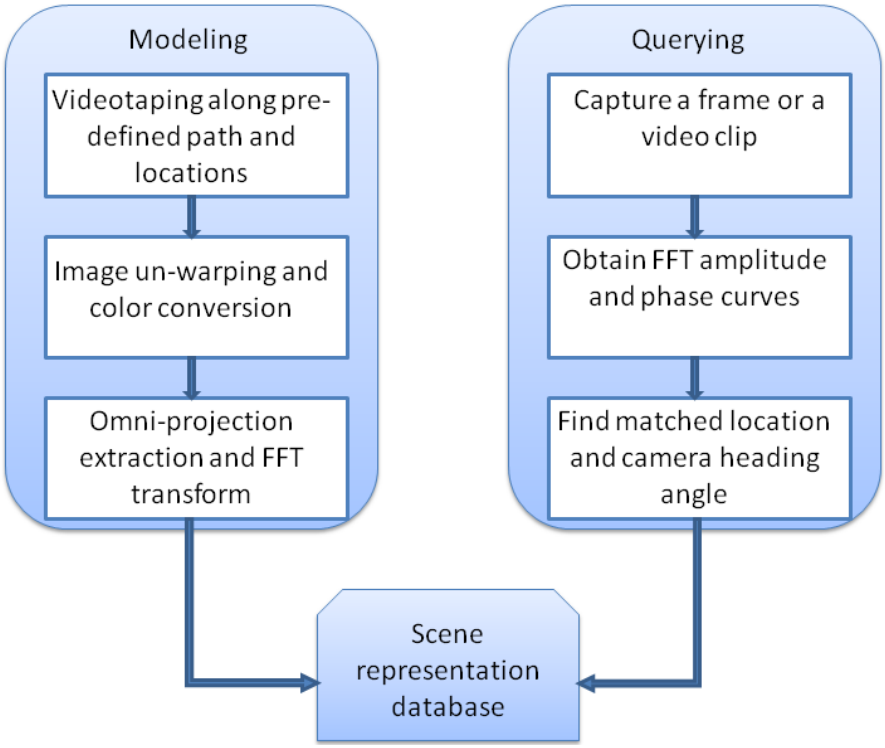


Figure 4.4: Work-flow of modeling and querying stages

where landmarks (e.g. door plate, corners) are located while these frames are captured, and then labeling the rest frames in between by interpolation in the defined floor plan coordinate system. In the testing stage, we sample the testing videos by selecting one from every 5-10 frames, to reduce the query computation while still make use of the multi-frame query advantage.

Using all the pixels in images of even a short video clip to represent a scene is too expensive for both communication and computation, and the data are not invariant to environment variables such as illumination or user heading orientation. Therefore, we propose a concise and effective representation for the omnidirectional images by using a number of one-dimensional omnidirectional projections (omni-projections) for each panoramic scene. It is observed that an indoor environment often has plenty of vertical lines (door edges, pillar edges, notice boards, windows, etc.), and the distribution of these lines around the user provides unique landmarks for localization. However for computational efficiency as well as robustness, we do not extract those line features directly; instead we embed them inside of

the proposed omni-projection representations, even though these line features can be utilized in the future with a correlation-based matching method to estimate a viewer's location.

Newly extracted features of an input image are used as query keys to localize and navigate in the environment represented by the database. Because the omni-projection presentation will be different even at the same location if the smart phone is not held vertically, we develop a function in the database acquisition and testing procedure by extracting the smart phone's built-in gyroscope values, and give the developer or the user a hint if the phone is not held properly. More details are discussed in Section 4.3.2.

Many man-made structures and objects look very similar, particularly for indoor environments, so it is a challenging task for accurate localization using only single frame. We therefore adopt a multiple frame querying mechanism by extracting a sequence of omni-projection features from a short video clip, which can greatly reduce the probability of false positive indexing. When the database scales up for a large scene, it is very time consuming to sequentially match the input frame features with all the images in the database of the large scene. So we use General Purpose GPU (GPGPU) to parallelize the query procedure and accelerate the querying speed, the details of which are shown in Section 4.2.

Like any other image-based localization method, occlusion by moving persons and scene changes due to varying illuminations will influence the scene representation and thus challenge the effectiveness of localization systems. In extreme cases, if the scene representations are identical with each other (e.g. empty rooms with uniform-colored walls), or there are large self-occlusions (e.g. camera is blocked by users hand), using appearance-based image localization system alone will have challenging problems. Our system has two advantages over other approaches: (1) the omni-lens has a 360×82 degrees field of view, which captures wider areas visual information and is less vulnerable to local occlusions or scene changes if they only occupy in small part of an omnidirectional image; (2) It is not a single image, but a short video sequences captured while the user is walking is utilized, so even though one of

the frames has significant occlusions or changes, the rest frames can help to provide stable environment visual information for an effective localization.

The modeling procedure is space intensive both during the processing (with intermediate data and the final results) and the database storage. The querying procedure, on the other hand, could be very time consuming, particularly with multi-frame querying. The basic idea of multi-frames indexing is to use a sequence of newly captured video frames to query a pre-built video frame database to increase the success rate. Even with the preprocessing step (in the front end of an iPhone) to reduce the size of the input image data, the computational cost of multi-frame query in the database would be high if the database is too large. As a result, both modeling and querying procedures are designed to be manipulated in the cloud server part, where parallel processing can be applied for acceleration.

One parallel strategy is to partition the input video into individual frames and query the database with each frame in parallel. Then, for every input frame, we compare it with all the frames in database one-by-one. After all the queries return their matching candidates, which for example, top 5 matches, an aggregation step can be obtained by assuming that both the input and the matching sequences are temporal sequences. In this way, the querying process could have four levels of parallelisms. First, we divide the search database into multiple subspaces, for example, each floor data is counted as one subspace, and all the subspaces can be parallelly searched with CUDA streams. Second, within each subspace, we process all the frames of an input query video clip in parallel. Third, we use multiple threads to compare each input frame with multiple database frames simultaneously, rather than comparing with them one-by-one. Fourth, some of the operations in obtaining rotation-invariant projection curves, such as the Fourier transform algorithm can take advantage of the parallel processing. For doing this, the whole original omni-projection curves will be sent from the front end to the server, which is still very efficient in communication due to their low dimensionality.

The rest of the Chapter is organized as follows. Section 4.1 illustrates the image preprocessing and feature extraction procedure. Section 4.2 discusses the localization by indexing

approach, the multi-frame aggregation algorithm as well as issues in parallel processing. Section 4.3 provides real data experimental results with both accuracy and time performance analysis.

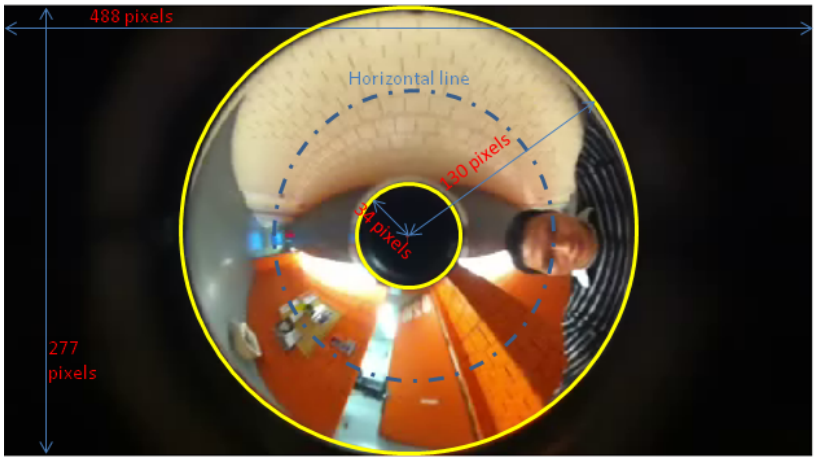
4.1 Preprocessing and Omni-Feature extraction

The original image frames captured by the GoPano lens and smart phone camera are fish-eye-like distorted images, as shown in Fig. 4.5a, which has to be rectified. Fig. 4.5c shows the un-warped image in a cylindrical projection representation, which has planar perspective projection in the vertical direction and spherical projection in the horizontal direction. For achieving this, a calibration procedure is needed to obtain all the required camera parameters which will be discussed in 4.1.1. After obtaining the preprocessed image, we can extract one-dimensional omnidirectional feature from it; we will discuss this in 4.1.2.

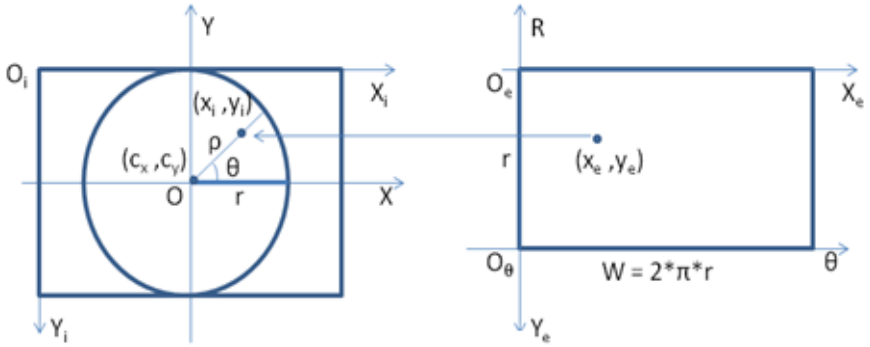
4.1.1 Calibration and preprocessing

There are many omnidirectional image calibration algorithms in literature [87][69][41], but the majority of them require heavy computation, which is not practical in real-time technology especially if implemented on a smart phone with limited computation resources. In our work, we propose a simple but effective calibration approach. We assume that the camera is held up-right and the lens' optical axis is horizontal, so the relationship between the original image and un-warped image can be illustrated in Fig. 4.5b [86][37]. When the camera is not held up-right, we will reminder the user to correct it by detecting the built-in IMU data, specifically, the tilt angle (see 4.3.2 for more details about this).

Denote the original pixel coordinate system as $X_iO_iY_i$, the un-warped image coordinate system as $X_eO_eY_e$, and the original circular image center as (C_x, C_y), then a pixel (x_e, y_e) in the un-warped image and the corresponding pixel (x_i, y_i) in the original image can be modeled as



(a)



(b)

(c)

Figure 4.5: (a) Original video frame and its parameters. (b) Geometric relationship between the original and un-warped images. (c) The un-warped omnidirectional image

$$\begin{cases} x_i = (r - y_e)cos(x_e \times \frac{2\pi}{W}) + C_x \\ y_i = (r - y_e)sin(x_e \times \frac{2\pi}{W}) + C_y \end{cases} \tag{4.1}$$

where r is the radius of the outer circle of the original circular image—the vertical dimension (in pixels) of the cylindrical image, and $W = 2\pi r$ is the perimeter of the circle, turning to the horizontal dimension (in pixels) of the cylindrical image.

This un-warping process is applied to every frame in the database and all the input query frames. Since such un-warped images still have distortion in the vertical direction (radial direction in the original images) due to the nonlinearity of the GoPano lens, we perform an image rectification step using a calibration target with known 3D information to correct the radial direction so that the projection in the vertical direction of the un-warped cylindrical images is a linear perspective projection [110]. By doing this, the effective focal length in the vertical direction is also found. From this point on, we assume that the image coordinates (u, v) are rectified from (x_e, y_e) , the u direction (horizontal direction) represents the 360-degrees panoramic view, and the v direction (vertical direction) is perspective.

4.1.2 Omni-feature extraction

We do not directly use the pixel values of the frames to query with the database. Instead, we extract rotation invariant features (denote it as Omni-Features) from the omni-projection curves, for both images' Hue, Saturation, and Intensity (HSI) channels and their respective gradients. In this subsection, we will discuss the detailed procedure of the omni-feature extraction for intensity channels, as shown in Algorithm 1.

According to our observation, when a visually impaired person capturing an omnidirectional image by holding the imaging system, the vertical lines usually appear in the middle of an image, while the topmost and bottommost part are ceilings and floors—such as the central black area in Fig. 4.5a—which do not provide valid information. Therefore, we denote the middle area as the Region of Interest (ROI), and generate our omni-features by only using the ROI. Currently rectangular ROI is manually determined to be 10% of the top

and bottom image area to remove useless information and also lower the memory usage as well as transmitting overload.

Algorithm 1: Omni-feature extraction

Data: An omnidirectional video captured by an iPhone

Result: Each frame's vector representation of the omni-feature *omniVector*

```
initialization;
GoPano image calibration;
generate a look-up table Tab() between oriImage and unDistortedImage;
while hasNextFrame() do
    for each pixel (u, v) in unDistortedImage do
        calculate unDistortedImage(u,v) value using Tab();
    end
    iImage = rgb2hsi(unDistortedImage);
    gImage = horizontalGradient(iImage);
    pVector = verticalProjection(gImage);
    npVector = normalize(pVector);
    fVector = FFT(npVector);
    omniVector = selectPrincipleComponent(fVector));
end
```

How to represent a scene with unique and concise features using the omnidirectional images? In our work, to represent a scene, six omni-projection curves are extracted from each corresponding cylindrical image channels, as shown in Fig. 4.6. Three of them are from Hue, Saturation and Intensity (HSI) channels of the image, and the remaining three are from the gradient magnitudes of the Hue, Saturation and Intensity channels. Denote the original image of the HSI channels as $f(u, v)$, where u and v stand for the horizontal and vertical directions respectively. The gradient magnitude image of $f(u, v)$ is calculated as follows

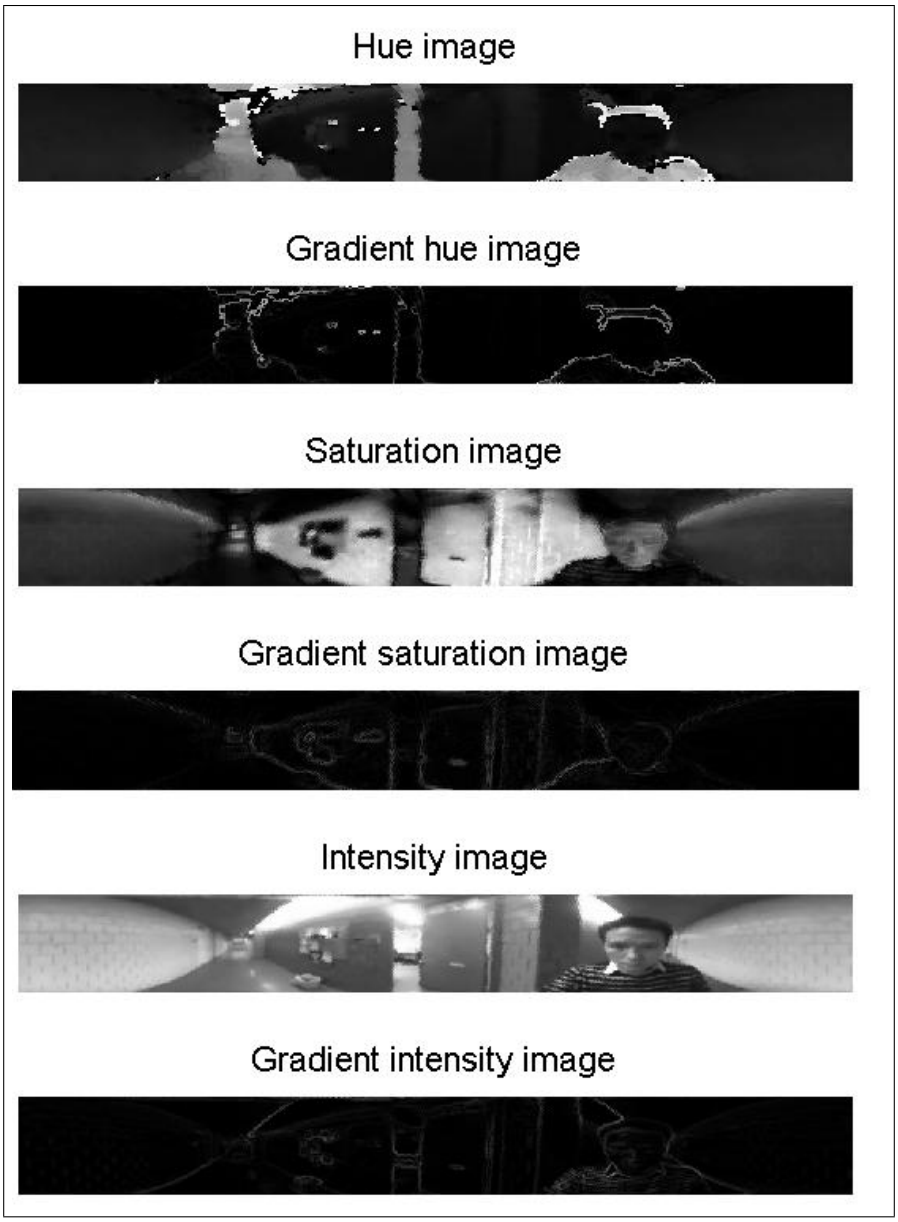


Figure 4.6: Visualization of the Hue, Saturation, Intensity (HSI) channels and their corresponding gradient images

$$|\nabla f(u,v)| = |\frac{\delta f}{\delta u}du + \frac{\delta f}{\delta v}dv| \tag{4.2}$$

where f can be H, S and I channels, $\frac{\delta f}{\delta u}$ is gradient in the u direction, and $\frac{\delta f}{\delta v}$ is gradient in the v direction. L_2 norm of the $(\frac{\delta f}{\delta u}, \frac{\delta f}{\delta v})$ is used to calculate the magnitude. In practice,

since we mainly focus on the vertical lines—horizontal changes of the images, we only need to calculate the horizontal gradient.

We further define function g(u,v) as the image function to represent one of the six types of images (H, S, I and their gradient magnitudes, or simply gradients). The omni-projection curve $c(u)$ of the feature function $g(u, v)$ is generated by projecting the ROI of the image in the v direction:

$$c(u) = \sum_{v=0}^{H-1} g(u, v), u = 0, 1, 2, ..., W - 1 \tag{4.3}$$

where W and H are the image width and height of the ROI (where H is smaller than r), and $u = 0, 1, 2, ..., W - 1$ are the horizontal pixel indexes (corresponding angles from 0 to 360 degrees). As we can see the curve $c(u)$ is an one dimensional omnidirectional projection curve.

A linear normalization is applied to all the curves to turn them into the same scale and increase the robustness for the illumination changes.

With all the six curves, we store each and every one of them into environment database, and use them to compare with the new input curves for finding the optimal location for a newly input frame.

The omni-feature extraction algorithm is summarized in Algorithm 1. In the calibration step (Step 2), the camera is calibrated by finding the image center's position, the outside and inside radii of the circular images. In Step 3, a look-up table is built up between the original image *oriImage* and the undistorted image *unDistortedImage* for fast processing, using the geometric relationship between this two images, as shown in Fig. 4.5(b). Then, for each frame, we generate the undistorted image (Step 5), convert it from RGB color space to HSI space (Step 8), and select the intensity channel *iImage*. After that, we apply gradient operation to the *iImage* (Step 9), and project the resulting image onto an omnidirectional vector (Step 10). Finally, we normalize the vector (Step 11), transfer it into Fourier domain (Step 12), and generate the vector representation of the omni-feature by selecting the princi-

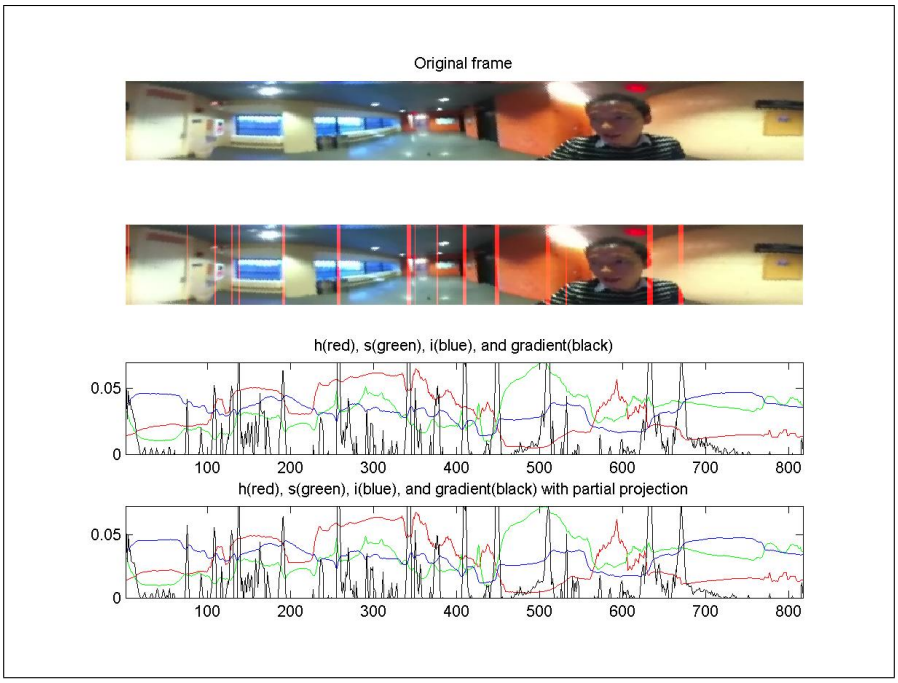


Figure 4.7: A sample omnidirectional frame and detected vertical lines

ple component (the first half of the vector) of the FFT result for reducing transmitting and storage load (Step 13).

Note that our work does not directly use individual vertical lines to represent or match scene images. Instead, we use a global feature vector to represent each panoramic scene image, and then match the pre-built scene omni-feature database extracted using the same feature extraction method. Our features, however, do relate to vertical lines of the scenes, because after we get the omni-features, these line features are included, as shown in Fig. 4.7. More result is shown in this video [3] to show the relationship between omni-features and vertical lines by real-time detecting vertical lines from our omni-features. Since the vertical line detecting algorithm is off the topic and not related with our global feature based localization purpose, we do not include it here, however, it could be used as an alternative local feature-based localization method in the future.

[3] https://youtu.be/MgGy_qt6I-Y

4.2 Localization by indexing

Localization service is critical to a visually impaired person's normal life, not only because it provides with the current position and orientation of a user, but also because it could supply additional information of the environment, e.g. locations of doors, positions of door-plates, which are very useful for them to judge and make decisions. The essential idea of omnidirectional imaging based localization is that we employ for each scene a unique and distinguishable feature, so each scene position is indexed by the omnidirectional feature. In this section, we will introduce in detail how to localize a user via indexing using our omni-features for visually impaired people.

4.2.1 Rotation-invariant feature extraction and rotation estimation

One question raised is that what if in the query stage, the images acquired at the same location, have a different heading direction. In this case, even though two omnidirectional images are captured at the identical location, they have different image contents. We prove that our omni-feature is rotation-invariant and thus have the ability to represent a scene no matter what heading direction is used when capturing this image up to a shifting angle.

For an arbitrary omni-projection curve $c(u), u = 0, 1, 2, ..., W-1$, if the camera rotates around the vertical axis, it will cause a circular shift of the cylindrical representation of the omnidirectional image, which then corresponds to a circular shift to the signal $c(u)$. If an omnidirectional image has a circular shift of u_0 to the right, this is equivalent to rotating the camera coordinate around z axes for $\Phi = -2\pi u_0/N$ [111]. Suppose the signal after a right circular shift u_0 is $c'(u)$, we have the following equation:

$$c'(u) = c(u - u_0) \tag{4.4}$$

Denote the FFT of $c(u)$ as a_k, the FFT of $c'(u)$ as b_k, then

$$\begin{cases} a_k = \sum_{u=0}^{W-1} x(u) e^{-je2\pi ku/W}, k = 0, 1, ..., W-1 \\ b_k = a_k e^{-je2\pi ku_0/W}, k = 0, 1, ..., W-1 \end{cases} \tag{4.5}$$

As we can see from the second equation of Equation 4.5, the magnitudes of the omni-projection curves are rotation-invariant, i.e., $|a_k| = |b_k|$. This is also why we use the FFT magnitudes of the six omni-projection curves of a query frame to do indexing in the database for avoiding the heading direction problem.

Utilizing the whole amplitude curve is much more economical in terms of storage used compared with storing the whole image pixels. We further improve this by only selecting the first half of the FFT magnitudes of each curve $|a_k|$ for indexing without losing localization accuracy by taking advantage of the symmetry property of the FFT transformation for discrete signals.

How we can then find the location and heading direction using the above feature? When a new omni-feature is extracted and we want to find its corresponding scene location, the distances of the six curves between the query and each database frame are calculated. The final match is the database frame that yields the smallest L_2 distance. From our experiments, we have found that the gradient of intensity curve provides the best discrimination. Once the correct location is determined with the above approach, we can further calculate the heading direction of the new omnidirectional input. We formulate the problem of finding the rotation angle of the current image (i.e. the amount of circular shift, or the heading direction of the image) to finding the maximal value of the circular correlation function (CCF)

$$CCF(u_0) = \sum_{u=0}^{W-1} c(u) \times c(u - u_0) \tag{4.6}$$

where $u_0 = 0, 1, ..., W-1$. According to the correlation theorem, we can calculate $CCF(u_0)$ as

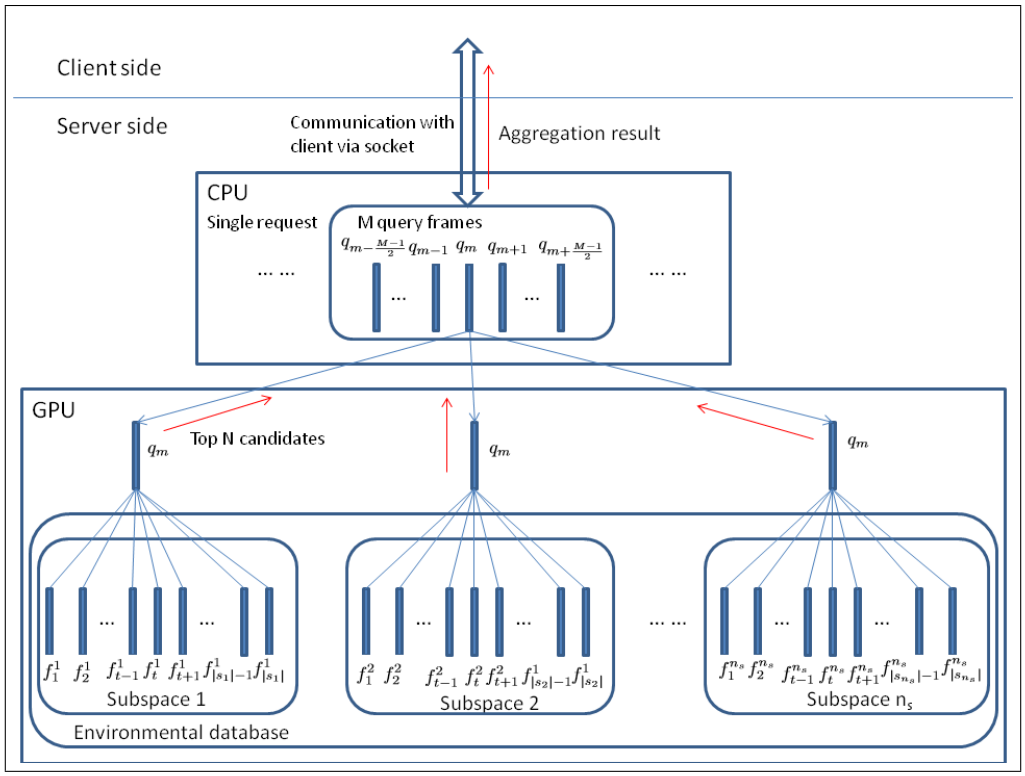


Figure 4.8: Parallel query with GPU in the server side

$$CCF(u_0) = F^{-1}\{a^*(k)b(k)\} \tag{4.7}$$

where F^{-1} is inverse Fourier transformation operator, and * is the conjugate operator.

Now we are able to find both the location and heading direction of a new input image. There are, however, still more issues left. In the real environment, because of the existence of the noises, the localization result using one image is not always accurate and not stable. In the following subsections, we will talk about how to solve this problem by using multiple images and aggregation techniques.

4.2.2 Multi-frame parallel indexing and median filtering

To increase the robustness of the localization algorithm, using a single input omnidirectional image is not sufficient; we need to input a sequence of images to query the environment

database for accurate localization. Each input image returns an optimal location, and all the returned locations are aggregated to generate a final location result.

When multiple frames are utilized, querying them with normal CPU is not efficient, therefore, we utilize GPU along with corresponding parallel algorithms to ensure a real-time performance. Fig. 4.8 shows a diagram of our GPU parallel searching algorithm on the server side. When a client sends a localization request of a frame q_m, it actually sends $\frac{M-1}{2}$ frames before the frame q_m, and $\frac{M-1}{2}$ frames after the frame q_m to the server (e.g. $M = 3$ or 5). The CPU of the server will copy each frame's feature to the GPU, and receive the top N candidates calculated by the GPUs for each frame after processing.

The environmental database is also divided into n_s subspaces (e.g., each floor of a building's environmental data can be stored as separate subspaces), and the n_i subspace has $|n_i|$ modeled frames. All the subspaces therefore can be searched in parallel with GPU, instead of searching all the subspaces sequentially with a single CPU.

Take the frame q_m as an example. The frame q_m is compared with all the frames f_t^i (t = 1, 2, ..., $|n_i|$) within each subspace i (i = 1, 2, ..., n_s). The same searching procedure is carried out to all frames also. Finally, all the returned candidates of the M query frames are aggregated and return to the client via TCP/IP socket.

To aggregate the return candidates, we can take advantage of the fact that both the query frames and the database frames are sequentially indexed, so we use a simple median filtering approach for the temporal aggregation.

We formalize our algorithm of median-filter-based parallel indexing and show it in Algorithm 2. In the initialization step, the server receives the data from the client and prepare for querying. In Step 2, for each query, the server will not only select the current testing frame, but also multiple other frames near the testing frame. Then, all the selected frames are copied to each CUDA stream one by one (Step 5), where the database of each subspace are stored and computation is taken care of concurrently. Within each stream, there are multiple threads, e.g. 32×256 threads, and each thread is responsible for the comparison of

the querying frame and database frame (Step 7). Streams are executed simultaneously, and all the threads within each stream are also carried out in parallel. CUDA synchronization is called to make sure all the tasks within each thread (Step 9) and all the work in each stream (Step 12) are finished before the program moves to the next step. Finally, the GPU collects the returned candidates, which could be used for the median-filter-based aggregation (Step 14).

Algorithm 2: GPU multi-stream paralleling indexing and median filtering

```
   Data: M query frames received from CPU
   Result: Location candidates FinalResult calculated by GPU
 1 initialization;
 2 selectNearbyFrames();
 3 for each selected frame q_m do
 4 |   for each CUDA stream do
 5 |   |   memoryCopyFrmoCPUtoGPU(q_m);
 6 |   |   for each CUDA thread do
 7 |   |   |   calculateDistance(q_m, f_i);
 8 |   |   end
 9 |   |   synchronizeThreads();
10 |   end
11 |   synchronizeStream();
12 |   Results = selectTopCandidates();
13 |   memoryCopyFromGPUtoCPU(Results);
14 |   FinalResult = medianFiltering (Results);
15 end
```

4.2.3 2D multi-frame aggregation based on candidates' densities

For majority of the indoor environments, such as normal corridors, rooms, the environment is modeled by walking through it once, and the above median-filter-based aggregation algorithm works well. However, for the environment with wide corridors or large rooms, modeling with a single traverse is not sufficient and we need multiple traversals (paths) to be able to cover the complete area and provide accurate result. Also, for a multi-frame based localization query, the query frame is sent to the server together with a number of frames before and after it. After receiving a large amount of position candidates for each query frame, a

more sophisticated aggregation algorithm is needed to integrate all these candidates together before we deliver a finalized location coordinate to the user. In this subsection, we will talk about these two problems in detail.

We demonstrate the application of the system for the areas such as corridors or rooms, which are large, by increasing the densities of the environmental modeling with multiple paths of video capture. Instead of capturing a video on only one path, we use multiple parallel paths to build the environment model. After that, we correlate the paths with the physical space coordinates by performing associated frame labeling and interpolation. Then, the final location is obtained (using Algorithm 3) by aggregating the possible candidates in the 2D coordinate plane.

If there is no noise, in an ideal case, for any input testing frame, all the returned candidates from the server should be at the exact the same position where the testing frame was captured, since the testing frame and the modeling frames (who generate the candidates) shall capture identical environmental information. However, because of various reasons (noise, scene similarity, etc.), some of the candidates may be disturbed and are far away from the ground truth position, while the majority of the candidates cluster around the ground truth area. In this thesis, according to the above observation, we have designed and implemented a 2D multi-frame aggregation algorithm by utilizing the densities of the candidates' distribution, and calculate the most likely finalized location estimation.

The algorithm is shown in Algorithm 3. In the initialization step, the candidates' indexes in the modeling frames are mapped to their actual floor plan 2D coordinates by checking the geo-referenced mapping table pre-generated while modeling the environment.

Then in Step 2, we count each tile's candidates number by a 2D binning operation, and obtain the query frame's location distribution array $locationDistriArray$. We sort this array in a descending order in Step 5, and obtain the first $TopC$ number (currently $TopC = 10$) of tiles in Step 6.

Algorithm 3: 2D multi-frame localization candidates aggregation

Data: Location candidates on the floor plan coordinate system
Result: Aggregated final location coordinate $FinalPosition$

```
initialization;
for each candidate C_i do
    Accumulate locationDistriArray(C_i);
end
rankLocationDistribution(locationDistriArray);
TopC = findTopDensityArea(locationDistriArray);
for index < NumOfTopC do
    if topDensityArea(index) > NumOfAllCandidates × tolerPer then
        FinalPosition = topDensityArea(index);
        break;
    end
end
```

In a perfect case, the top one tile of these $TopC$ candidates should be the best estimated result since it has the largest amount of candidates dropped in. This is, however, not always the case, because the tile with the most candidates may be a false positive estimation, and its nearby tiles may have very few candidates.

To increase the robustness of the estimation, in Step 7, we set a tolerant circle around each tile, with some radius (in our experiment, this radius is 3 meter.). Afterward, we count the total number of candidates within this circle. If the amount is greater than a threshold percent— $tolerPer$ (e.g. $tolerPer = 20\%$) of the total amount of candidates, the corresponding tile is estimated as the final position, and is returned to the user via the sockets. We will show the experiments with real data in the following sections.

4.3 Real data experiments: accuracy and time performance

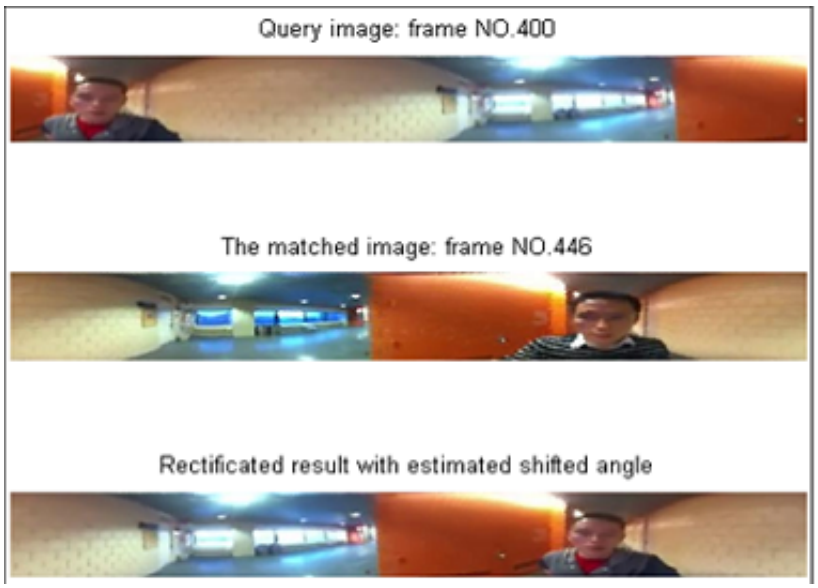
A number of experiments are carried out for testing the accuracy and time performance of the omnidirectional-vision-based localization approach, the details of which are provided in the following subsections.

4.3.1 Comparison of single-versus-multiple frame indexing

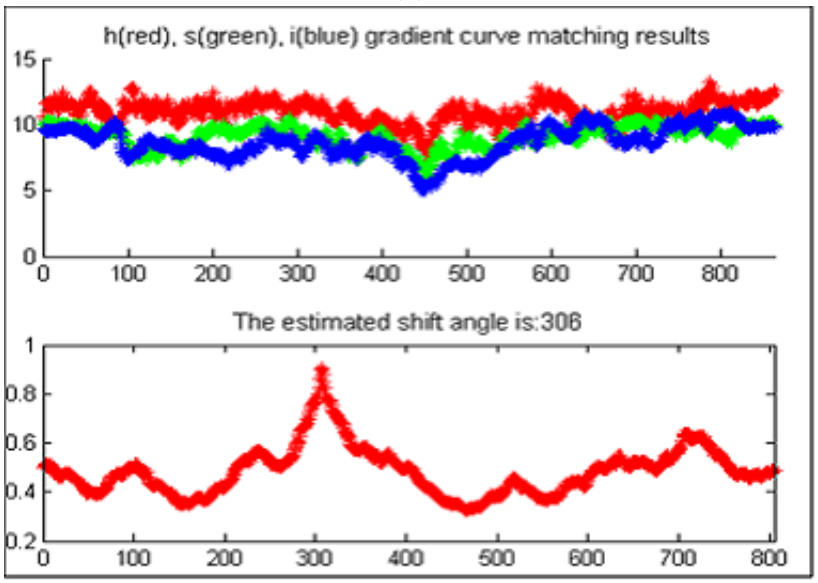
In the first experiment, we compare the accuracy of single-versus-multiple frame indexing with a small database of 862 frames modeling the environment with one single path.

One example of indexing a new query frame is shown in Fig. 4.9a and Fig. 4.9b. In Fig. 4.9a, we show an input frame, the target frame, and the shifted version of the input image after finding the heading angle difference respectively. In Fig. 4.9b, the first plot shows the searching results of the input frame with all the frames in the database using hue (red), saturation (green) and intensity (blue). We found that the intensity feature performs the best. The circular correlation function curve is shown in the second plot of Fig. 4.9b, indicating the heading angle difference between the input frame and the matched frame.

Because different scenes may have very similar omni-projection curve features, a query with only one single frame may cause false matches, as shown in Fig. 4.10 (top). In this figure, the horizontal axis is the index of input frames (of a new sequence) and the vertical axis is the index of database frames (of the old sequence). Both sequences cover the same area but are captured at different dates. The black curve shows the ground truth data of matching results, and the red curve shows the matching results by our system. As we can see, there are a few obvious mismatches around frame 150, 500, and 600 due to the scene similarities.



(a)



(b)

Figure 4.9: (a) An example of a query image and its matching result in a database. (b) The matching scores with database frames and the estimated heading differences between the query and database frames.

This leads us to design a multiple-frame approach: if we use a short sequence of input frames instead of just one single frame to perform the query, a temporally consistent match result for all the input frames will yield a much more robust result.

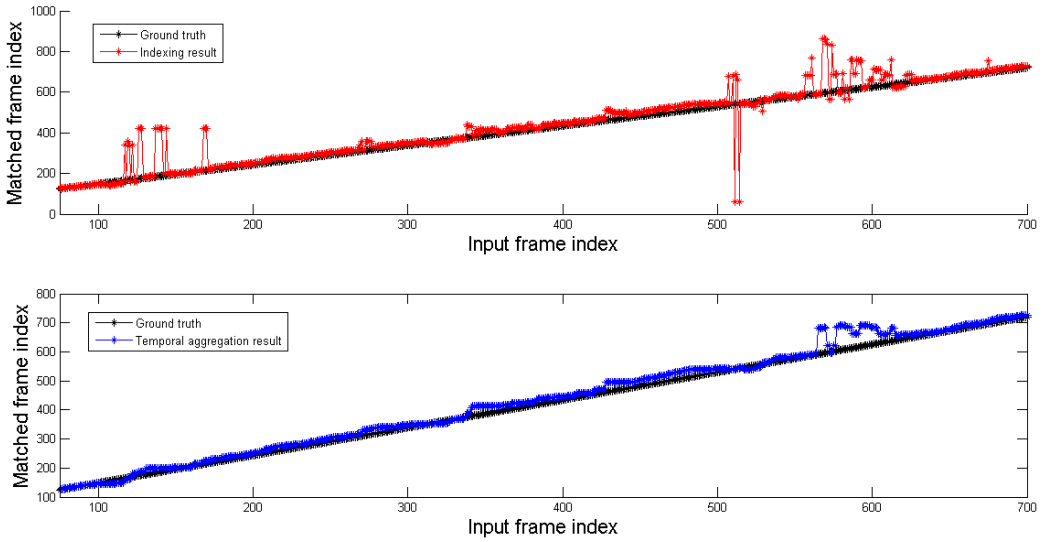


Figure 4.10: Matching results of all the test database frames without (top) and with temporal aggregation (bottom)

Fig. 4.10 (bottom) shows the testing results after the temporal aggregation. In this figure, for every frame, the querying results of its nearby 25 frames are aggregated and the median index value is used as the final result. We take advantage of the fact that both the query frames and the database frames are sequentially indexed, so we use a simple median filter of the 25 indexing results to obtain the temporal aggregation result. By a simple calculation, the average indexing error reduces to 14.7 frames with the simple temporal aggregation, from 25.3 frames with a single frame indexing. These correspond to 0.29m versus 0.51m distance error in space. As we can also see from the curve, temporal aggregation has corrected the very obvious mismatches and generated more robust results.

If using only a single CPU to search sequentially, the amount of time consumed would increase proportional to the number of frames in the database. Therefore we used GPUs to do the query in parallel, so that we can search all the frames and compare an input frame to multiple database frames at the same time, which will greatly reduce the time used. We demonstrate this by conducting an additional experiment. Fig. 4.11 shows the time used with and without many-core GPUs for a database with the number of images changed from

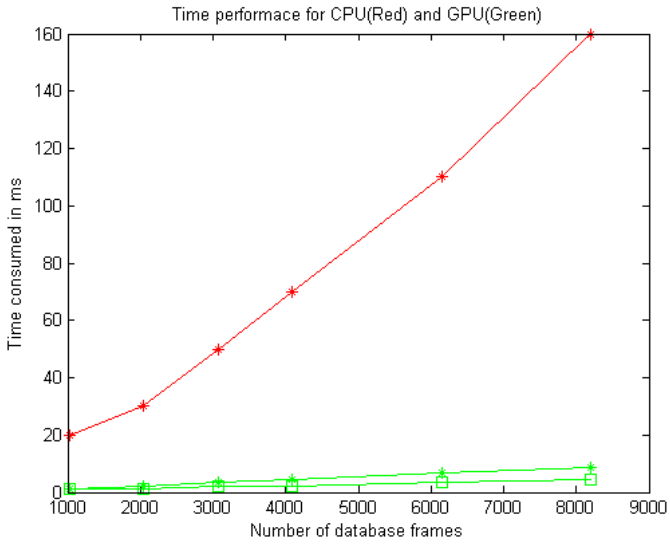


Figure 4.11: Time usage with and without GPU acceleration: Red without GPU; Green with GPUs. Curves with squares are experiments using 2048 threads while curves with asterisks are experiments using 1024 threads

1000 frames to 8000 frames. In the single CPU version, the time spent increases from 20 ms to 160 ms, whereas using a many-core GPU (Kepler K20 chip), the time is reduced by 20 times (from 1.13 ms to 8.82 ms, for databases of 1000 to 8000 frames). Note that this test only has a database with a few thousand frames. With a larger database of an indoor scene, the time spending on a single CPU will be prohibitive, whereas using multi-core CPUs/GPUs, the time spending can be greatly reduced.

4.3.2 Impact of tilt angles of the camera

Previous idea assumes that while capturing the omnidirectional image, the smart-phone and the omnidirectional lens are held vertically in both the environmental modeling and online querying procedure. Even though this assumption is easy to satisfy in most cases if we provide the visually impaired people a basic training before using the localization system, there are situations where the smart-phone may not hold perfectly vertically.

In this subsection, we further analyze the impact of the holding tilt angle problem quantitatively, and provide an effective solution via utilizing the smart-phone IMU data for detecting the tilt angle and using voice feedback to warn the users.

To ensure that the smart phone is held (almost) vertically in both the modeling and testing procedures, the tilt-correction function we have developed uses a simple, but effective method to detect the current smart phones tilt angle by checking the built-in gyroscope sensor. If the tilt angle is larger than a threshold (9 degrees in current setting), the smart phone will vibrate and hint the user to adjust the position [4]. With a little bit of training, the users can quickly adapt the correct way of holding the smart phone. In the future, we can even use the detected tilt angles to rectify the images if the angles are relatively small (below or close to the threshold).

If the camera lens is not pointing perpendicularly up, the omnidirectional features will change a lot, thus even two images captured in the same position have very different omni-features. Fig. 4.12 shows the matching results with and without tilt-correction function on, where the black lines in both curves show the ground truth locations in terms of the frame index, the red line on the top plot is the result with tilt-correction, and the blue line in the bottom plot is the result without tilt-correction. We can see from the plots that the accuracy increases significantly after the correction.

4.3.3 Localization in wide areas: 2D aggregation

To provide localization service in large areas, e.g. large rooms or wide corridor, single path modeling is not sufficient when the testing path is far away from the modeling path. In the third experiment, we model a whole floor with multiple paths (5 parallel paths), and test our Algorithm 3 for 2D aggregation result.

We first capture video sequences of the entire floor along these five paths as the training databases, one of which is shown in red line in Fig. 4.13. We then capture two other video

[4] https://youtu.be/02C17A2dpWI

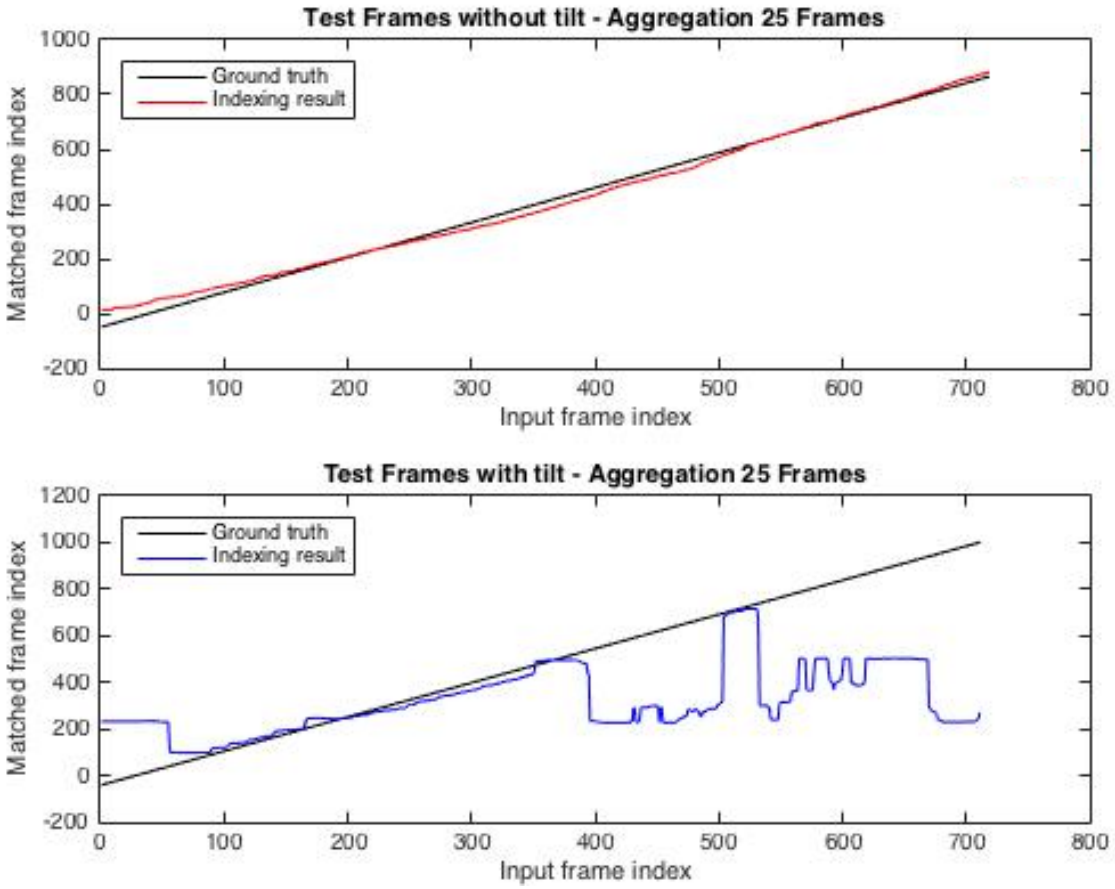


Figure 4.12: Matching results with and without tilt-correction function

sequences as the testing data sets, as shown in the blue and green lines, respectively. Some sample omnidirectional images used in the modeling process are also shown around the map in Fig. 4.13, as well as their geo-locations attached to the floor map.

For each testing frame, we query the entire training database and find the most similar stored features with the smallest Euclidean distance. Many scenes in the environment may be very similar, for example, the two images in Fig. **??**, so the returned feature with the smallest distance may not necessary be the real match. We then select the top N (N = 15 in this experiment) features as the candidates. Also, to solve the scene similarity problem, for each querying frame, we not only use itself as the query key, but we also use M-1 (M = 11) other frames, (M-1)/2 before and (M-1)/2 after this frame, as the query keys to generate candidates. We test all these frames within all the P (P = 5) paths, the frames of which are

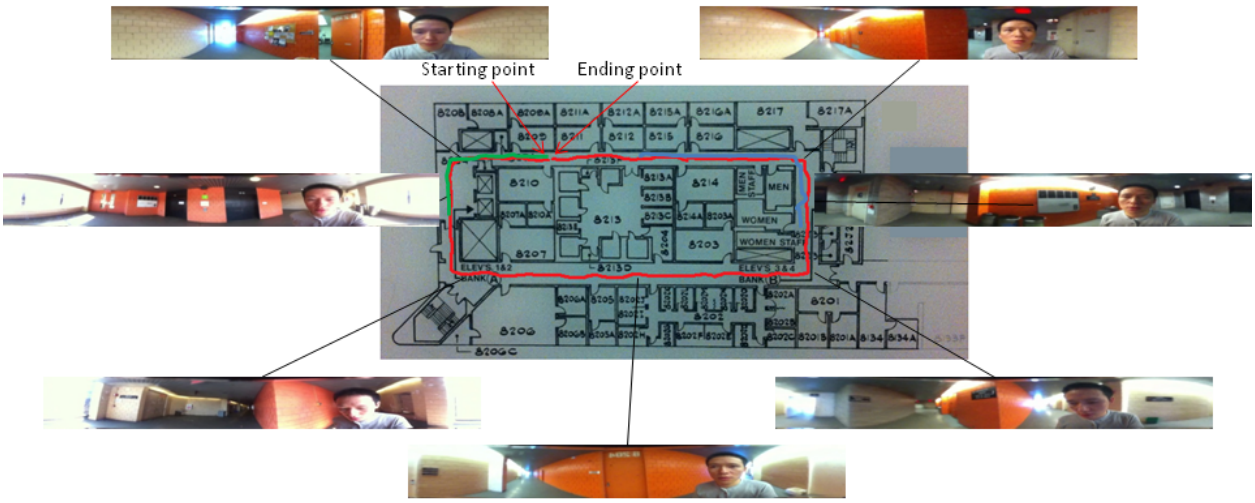


Figure 4.13: One of the eight floors testing environment and some sample omnidirectional images. The red line in the map is the modeled path. The blue and green lines are testing paths

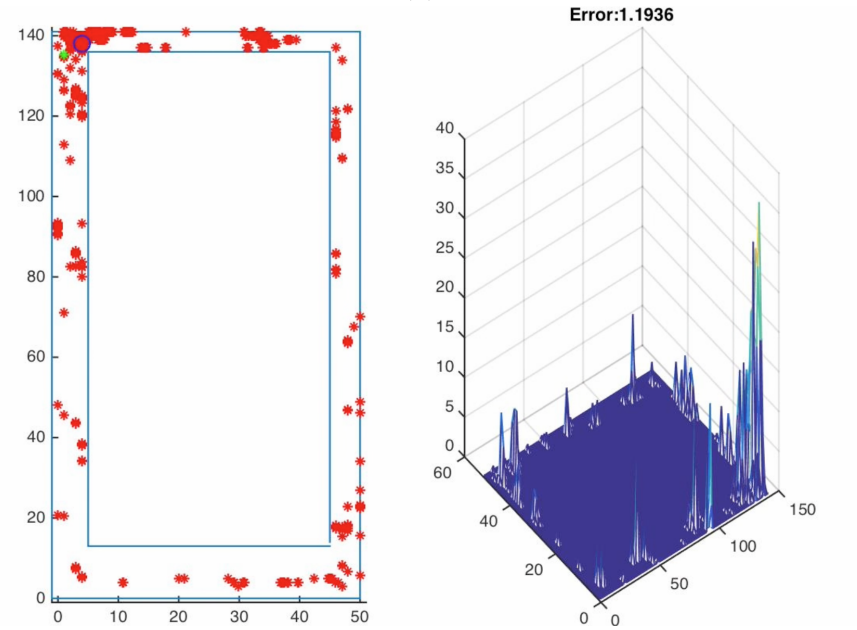
all labeled on the floor plan coordinate system, and retrieve all the candidates. Consequently, there are total $N \times M \times P$ candidates.

In Fig. 4.14b, the red stars are the 825 location candidates of a query frame and its related 10 frames. Note that several frames may return the same location, so each red star may represent more than one candidate. In the left image of Fig. 4.14b, the coordinate system has the same scale as the floor plan. Each unit in both horizontal and vertical directions corresponds 30cm in the physical space. The green star shows the ground truth location and the blue circle shows the final estimation of the testing frame. The distribution of the candidates is also illustrated in the right plot.

The x and y coordinates of right figure of Fig. 4.14b are the floor plan coordinates, and the height of each point represents the number of candidates falling into this position. Since all the modeling frames captured near the testing frame have similar contents and thus have alike features, the majority of the candidates drop near the location where the testing frame is captured.

In this example, as shown in Fig. 4.14b, the testing frame's ground truth location is on the top left corner of the floor plan (the green star), as we can see, the majority of the

(a)



(b) Aggregation visualization

Figure 4.14: (a) Two examples of very similar scenes. (b) Matching results of a frame and its neighbors against a multi-path database

candidates drop on the top corner area. The rightmost corner in the right figure of Fig. 4.14 also shows that the number of candidates (representing by the height) near the ground truth position is much larger than the rest of places. There are some false positives distributed around the whole floor as shown in both left and right figure. A short video clip can be found here [5] to visualize the robustness of the 2D aggregation algorithm with more queries. The number on the top right of Fig. 4.14 shows the estimated error in meters.

After the aggregation of all the candidates completes, we obtain the final candidate (the blue circle in Fig. 4.14), and use voice feedback to notify the user.

[5] https://youtu.be/gPGGcAfFsvY

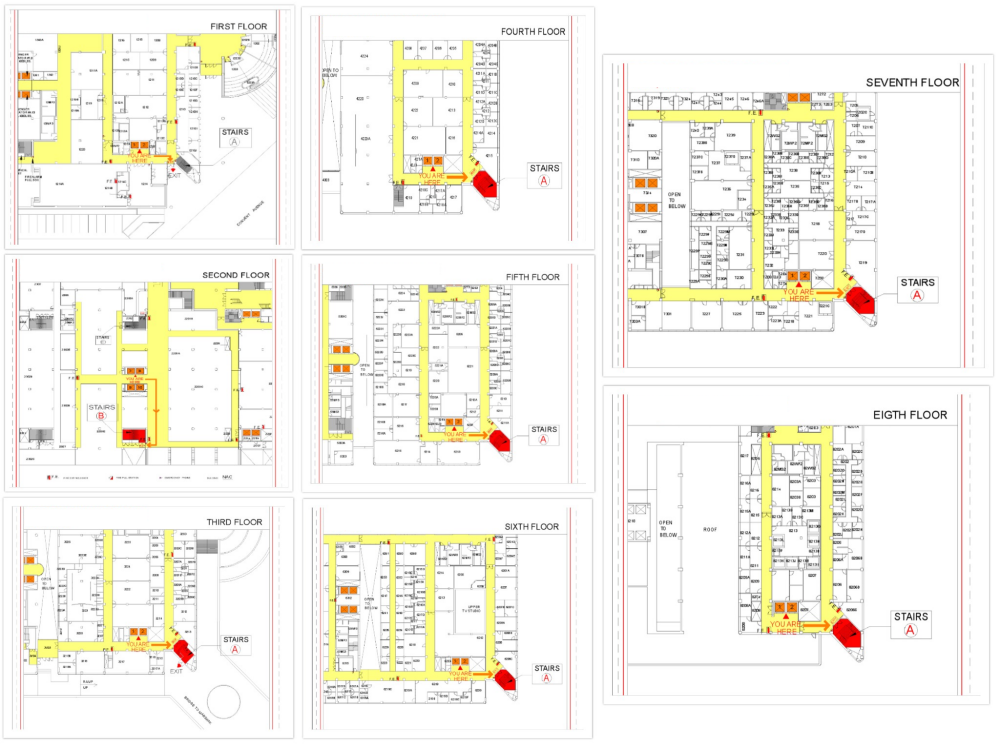


Figure 4.15: Floor plans of an eight-floor building

4.3.4 Time performance experiment

To show the time performance of the approach for a larger scale scene, in the fourth experiment, we extended the database from one floor of a campus building to the entire eight floors of the building. The floor plan of the building is shown in Fig. 4.15.

We held an iPhone mounted with a GoPano lens, and walked through each floor while recording the video at a normal walking pace. It took 32 minutes and there were 47617 frames in total for all the eight floors. Each floor (except the second floor) has an area of around 40 meters by 14 meters (around 560 square meters). We extracted features using the method illustrated in Fig. 4.4, and stored each floor's features into separate subspaces for later query operations.

As we discussed before, the hand-held working mode, instead of the head-mounted mode, is preferred by the users. Even though in this mode, the user's face is within the view of the camera, since it only occupies a small amount of omnidirectional images (less than 10

percent), and we use multiple images based query in the localization algorithm, it does not have significant influence on the generated features and the final localization error.

In the testing process, we selected one frame in every 5 frames for the testing, and constructed a testing data set. The total amount of testing frames is 9517 frames. These test frames are excluded from the training database so the testing frames and the database frames are not overlap with each other. We tested the time performance as well as the accuracy of our system at this campus building scale, and explored the performance of different aggregation results on different error tolerance threshold.

We assume that we do not know the users position at the very beginning, in order to make solution more general to all the cases. Even though applying some prior knowledge, such as initial positions, or using temporal information, may have some benefits, for example, reducing the searching database size, but it at the same time lowers down the usability of the system, since it requires the blind people need to ask others for the initial position, therefore reduces the users' independence. In the following experiments, we do not apply prior knowledge when querying the database.

Relating each frame with the floor plan, we can build a one-to-one mapping between the database frames and the walking path in the floor plan. Currently, only the frame's location within the floor plan is tagged, but it is also possible to relate the frames with more useful information, like office number, signs, etc.

By checking the mapping table and the error tolerance, we can estimate the localization error in terms of real geo-location distance. For this testing experiment, we used the difference between the original frame's index and the estimated index to evaluate the localization accuracy.

If the difference of the estimated index and the ground truth index were below a threshold, we labeled this query as accurate, as shown in Equation 4.8.

$$|I_{original} - I_{estimated}| \leq T_{threshold} \tag{4.8}$$

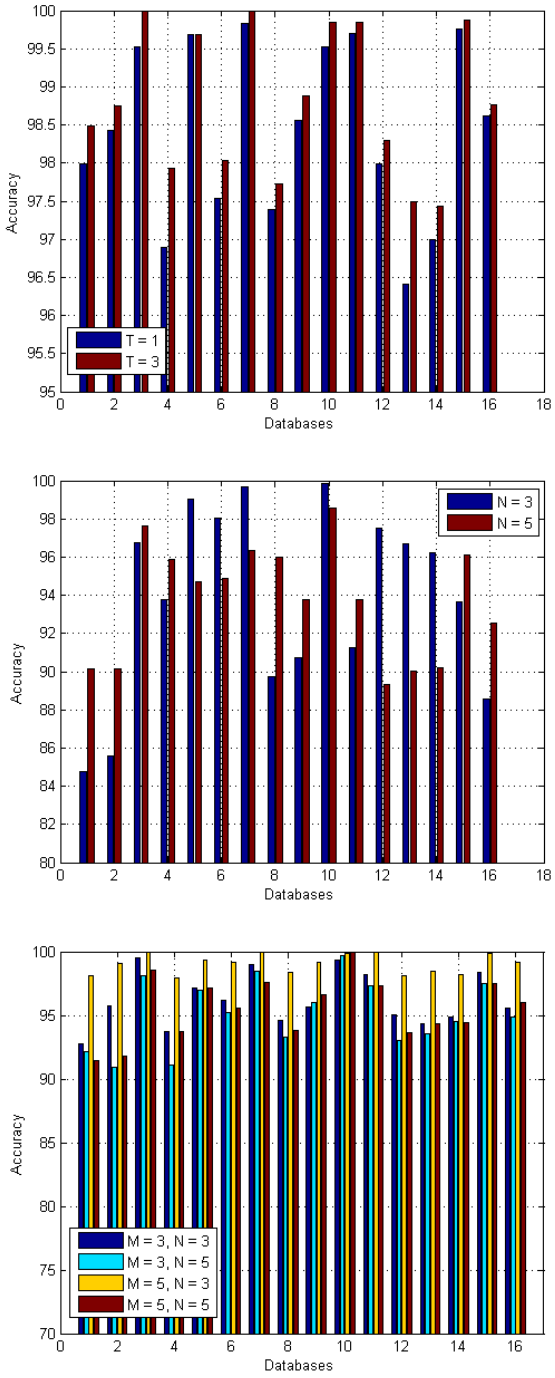
where $I_{original}$ is the original frame index number, and $I_{estimated}$ is the best matching result returned by GPU.

For each testing frame, we returned the top N (N = 3 or 5) candidates. To increase the robustness of the results, we used multiple frames nearby the current frame to query the database. Assume the number of querying frames is M (M = 3, 5), we use additional $\frac{M-1}{2}$ frames before the testing frame and $\frac{M+1}{2}$ frames after the testing frame to test the database.

For every query, there are $N \times M$ candidates, and we use median value aggregation to find the best result. Typically the few false results are outliers among a group of correct candidates. Therefore using a median filter among the $N \times M$ candidates, can remove the few outliers among the correct indexes values.

Fig. 4.16 shows the experimental results using the parallel searching strategies shown in Fig. 4.8 and the median-filtering based aggregation approach. The x axis stands for the feature databases for each floor, and the y axis stands for the localization accuracy. In Fig. 4.16(a), we tested the query accuracy with different threshold $T_{threshold}$ with the single-frame indexing (M=1) and a single return for each query (N=1). When the error tolerance threshold (T) increases from 1 frame (the blue bars) to 3 frames (the red bars), the accuracy increases on all the databases: on averaged the correct rate was increased from 98.43% to 98.82%. This shows that for over 50,000 images of the 8-floor building, the scene mostly doesn't repeat itself.

In Fig. 4.16a we tested various number of returns (N = 3 and 5) when doing single-frame query with a 3-frame indexing tolerance. We use the median value aggregation strategy to the N returned candidates for each query. When the number of candidates increase, as shown in Fig. 4.16a—the number of candidates increase from 3 to 5— only half the databases' accuracy increase. Further, to find the best accuracy performance, we tested a series of combination of the number of indexing frames M and the number of top returns N for each query, as shown in Fig. 4.16a, and the results show that when $M = 5$, and



(a)

Figure 4.16: (a) Single-frame indexing (M = 1), and single return (N = 1). (b) Single-frame indexing (M = 1) and a 3-frame indexing tolerance (T =3). (c) Finding the best numbers for indexing frames and returns when the indexing tolerance T = 3

$N = 3$, the system achieves the best accuracy: the average successful rate is 99.06% when the indexing error tolerance is 3 frames (T=3).

Experiments on the database of the eight-floor building demonstrated a real-time response (14fps). Each query uses around 70ms, among which about 40ms is consumed by the GPU searching, and 30ms is consumed by the socket communication. Note the current experiments have not fully used the capacity of the server. Since we can apply tens of thousands of threads in one or multiple GPUs, the acceleration rate has potential to improve.

Chapter 5

Refining Indoor Localization Using 3D Approaches

Omnidirectional imaging based indoor localization has the advantage of utilizing all the available visual information surrounding a visually impaired person for localizing themselves in the pre-built environmental models. However, in some cases, for example in high similarity scenes or where higher accuracy results are demanded, extra localization refinement is needed.

In this chapter, we focus on three possible refinement localization approaches, which are still based on visual information instead of other non-visual modalities. First, we test Structure from Motion (SfM) based indoor localization refinement method (Section 5.1). Then, we propose a multi-view omnidirectional geometry approach for localization refinement, which will be discussed in Section 5.2. Finally, we explore a direct 3D approach for acquiring and applying 3D environment information for the purpose of localization refinement, as discussed in Section 5.3. The overall strategy is that the omni-directional image based localization approach narrow the user's location to a small area, and then the 3D based approaches can be more effectively used in terms of both the computation time needed and the robustness of matches. In this chapter, we discuss the basic algorithms and present some preliminary

results. Future work is needed to fully integrate the refinement algorithms into a workable system for assistive navigation.

5.1 Structure-from-Motion (SfM) based localization refinement

The solution in Chapter 4 utilizes the completeness attribute of the visual information around a location, and uses the prior mapping knowledge between the omnidirectional image features and the floor plans for further retrieving locations for any given new images. An alternative approach of addressing the visual information based localization is to analyze the geometric relationship between the contents within an input image (typically with a normal field of view) and the environment 3D model. The 3D approach can be also used for refining the result from the 2D approach.

The major difference between 3D structure based localization and previous full Field of View based localization is that we explore the 3D geometric information—which were embedded into the image pixels during the imaging process, and try to utilize them to show the locations where these images are captured.

Structure from Motion is a technique used for obtaining the information about both the 3D motion and the geometry of the 3D scene viewed by a 2D image sequence. It is used in a wide range of applications such as photogrammetric survey [45], automatic reconstruction of virtual reality models from video sequences [112], camera motion estimation for augmented purpose so that computer-generated objects can be inserted into videos of real-world scenes [49], or improve the pose accuracy of Google Street View images [43]. Structure from motion approaches can also be used for accurate indoor localization. In our case, an initial estimation is already available from the 2D approach, so the 3D refinement can be more efficient.

Even though the basic principles are similar, there are several differences between Structure from motion (SfM) and Simultaneous Localization And Mapping (SLAM)—the problem

of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it, which is widely studied in robotics community. First, a traditional SLAM system usually relies on continuous (both space and time) sensing data acquired from one or several sensors for a successful mapping and localization, while for a SfM system, the visual information—images from different kinds of imaging systems, can be obtained from different time (e.g. could even be from different years [60]) and heterogeneous imaging system without having any constraints on the space-and-time order these images are captured. Second, the scale of a traditional SLAM system could be just a room, one floor of a building, or several floors within a building, since large scale outdoor localization is almost dominated by GPS-based methods, which makes larger scale outdoor SLAM unnecessary in majority of the cases, while the scale of SfM can be very large for variant of applications, for example, up to a city-scale, a country-scale or a planet-scale (e.g., the SfM model built using Google Street View images for the whole earth [43]).

The overall diagram is shown in Fig. 5.1. We divide the approach into two stages: offline 3D environmental modeling and online portable device localization. A 3D model of the environment with visual information attached is needed to be built beforehand for this refinement. A straightforward way of building 3D model is to use images taken in the environment and apply structure from motion technique. Subsection 5.1.1 will discuss the approach of creating an effective SfM model. After we have a model, we can analyze the principle of how the input image is formed, and apply geometric constraints based algorithm, such as pNp algorithm [53] to quantitatively calculate the camera pose within the environment model's coordinate system. Subsection 5.1.2 will discuss how to use pNp algorithm to perform pose estimation. Experiment results on real device and indoor rooms are carried out and also explained in subsection 5.1.3.

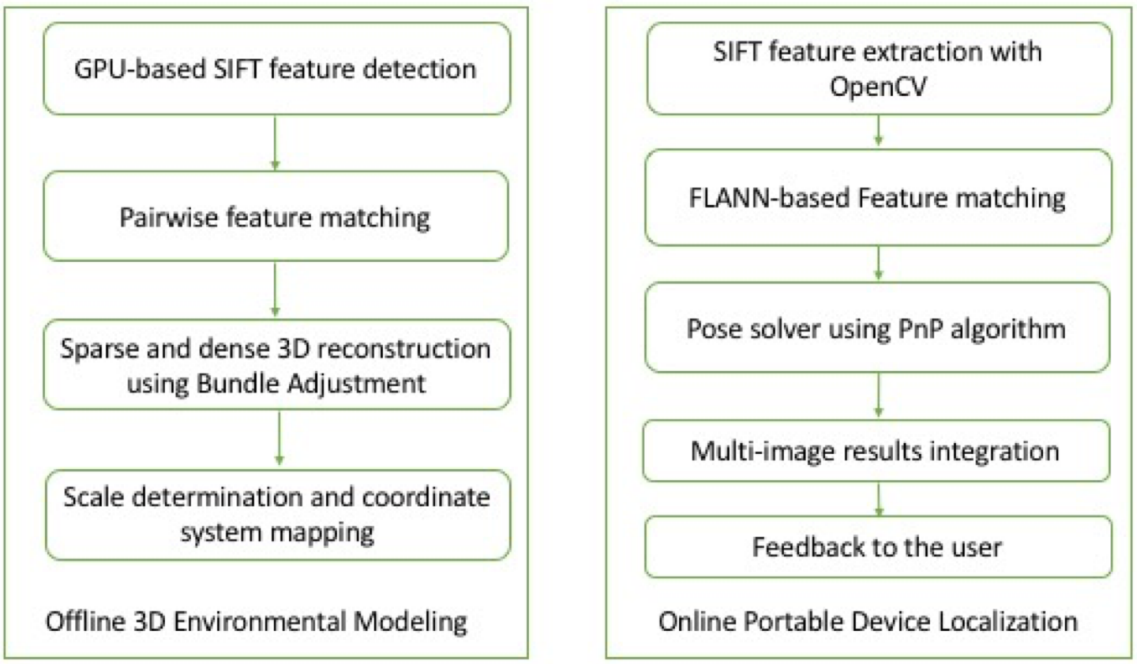


Figure 5.1: SfM based localization refinement diagram

5.1.1 Environmental modeling using SfM

In recent years, many SfM tools and software packages have been developed to generate 3D point clouds. Some of them are commercial while others are open sourced. The proprietary ones include Microsoft's Photosynth [1], Autodesk's 123D Catch [2], and 3DFlow's Zephyr [3]. The open source packages include Insight 3D [4], and SFMToolkit [5]. Others are from research groups, which include Bundler [72] by Noah Snavely, VisualSFM [99] by Changchang Wu, and OpenSLAM [78] by Kuemmerle *et al.* The popularity of SfM packages makes the task of generating a 3D point cloud of a scene easier and greatly facilitates SfM-based image localization research. In our work, we adopt the framework of the VisualSFM [99].

[1] http://photosynth.net/
[2] http://www.123dapp.com/catch
[3] http://www.3dflow.net/3df-zephyr-pro-3d-models-from-photos/
[4] http://insight3d.sourceforge.net/
[5] https://github.com/dddExperiments/SFMToolkit

The creation of the structure from motion model work in this thesis involves three parts: (1) GPU-based SIFT feature detection; (2) Pair-wise feature matching; (3) Sparse and dense 3D reconstruction using SBA.

(1) GPU-based SIFT feature detection

Scale-invariant feature transform (or SIFT) is an algorithm in computer vision to detect and describe local features in images, first proposed by Lowe [57]. There are a couple of open-source or free tools for extracting SIFT features for a given image [96][29][97], however, as the resolution of the images improves or the number of images increases, using Graphic Processing Units (GPU) to parallelize and accelerate feature detection procedure is attracting many researchers [14][100].

In our current experiments, we modeled two environments on our campus. One is an indoor room, as shown in Fig. 5.2(a), and the other is an outdoor building, as shown in Fig. 5.2(b). For the indoor model, we use 256 images, and for the outdoor model, we use 466 images. The first model is used in our Google Glass based localization experiment as the indoor environmental model. The second model is presented here as an illustration that even though our proposed refinement approach is intended for indoor environments, it can be naturally extended for outdoor scenes also. All the SIFT features of the images for these two models are extracted and stored for further process.

(a) (b)

Figure 5.2: (a) An indoor environment example. (b) An outdoor environment example

(2) Pairwise feature matching

Once image features are extracted and described for each and every image in the image sets that are used for reconstruct the 3D model, they need to be matched with each other to create the geometric constraints that reflect the positions and orientations relationship when these images are captured. Matching features among the images is one of the most time consuming steps in the SfM steps. If there are n images, it takes $O(n^2)$ time complexity to pairwise match them. Even though this step can be parallelized using GPU or multiple machines, since there are usually several thousands of features in a typical image, it is still a bottleneck to improve the speed of the whole procedure.

The matching process is carried out pairwise, and the matching results are stored as intermediate files. So if we want to scale up our environment by adding more images, we can expand the model incrementally without redo the feature extraction and matching process for a second time.

(3) Sparse and dense 3D reconstruction via Bundle Adjustment

Bundle Adjustment (BA) is the joint non-linear optimization of structure from motion (SfM) parameters, including structure of the environment and the camera's motion, at the same time, usually using Levenberg-Marquardt (LM) method [95][66]. It actually includes two problems: structure recovery problem and motion estimation problem, the connotation of which are described as follows:

Structure (scene geometry) problem: given a pair of 2D point matches in two or more images, where is the corresponding 3D point?

Motion (camera geometry) problem: given a set of 2D point matches in two or more images, what are the cameras pose matrices while taking the images?

Assume that there are m cameras and n points, denote X_i as the ith point, x_i is its corresponding point in 2D image, and P_j is the jth matrix containing the camera's parame-

ter. The bundle adjustment problem can be formalized as the minimization of reprojection function $E(P, X)$,

$$E(P, X) = \sum_{i=1}^{m} \sum_{j=1}^{n} Distance(x_{ij}, P_i X_j) \tag{5.1}$$

Once the structure and the motion are figured out, we can visualize them in terms of 3D point cloud and camera trajectory. Two example models for the indoor and outdoor environment is shown in Fig. 5.3a and Fig. 5.3b, which are correspondent 3D models of scenes in Fig. 5.2a and Fig. 5.2b.

The optimized camera parameters include structure information, in terms of 3D point cloud. However, the SfM model only represents the physical 3D space up to a scale. To solve this scale problem, we need extra knowledge about the environment, for example, the known distance between two points in the model.

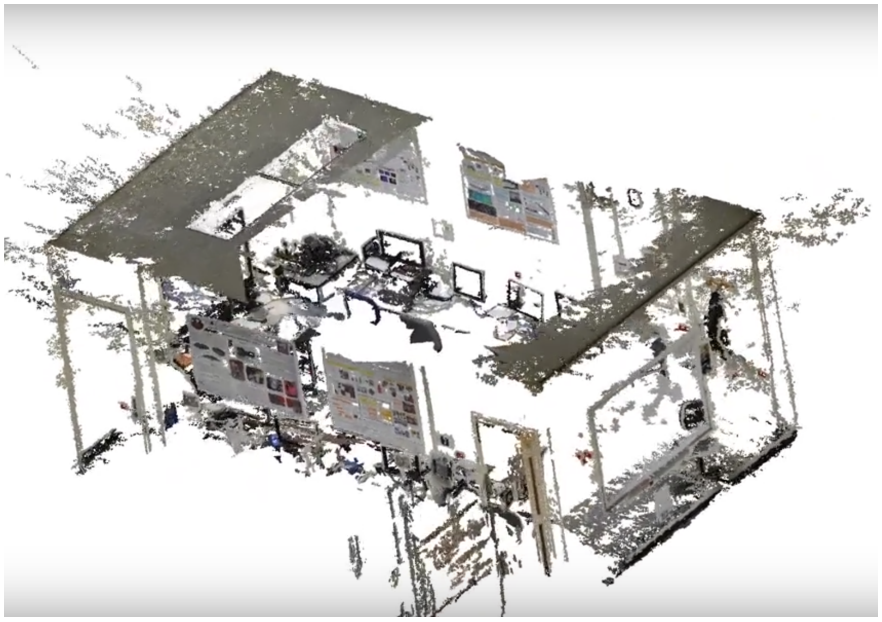
5.1.2 Visual computing for pose estimation

Once a model is created, new images can be captured within the same environment for localization query. In our work, Google Glass is used as the image data acquisition device, since it's both portable, hands-free and also supports voice command as well as feedback. This subsection discusses the detailed visual computing techniques used in the localization procedure.

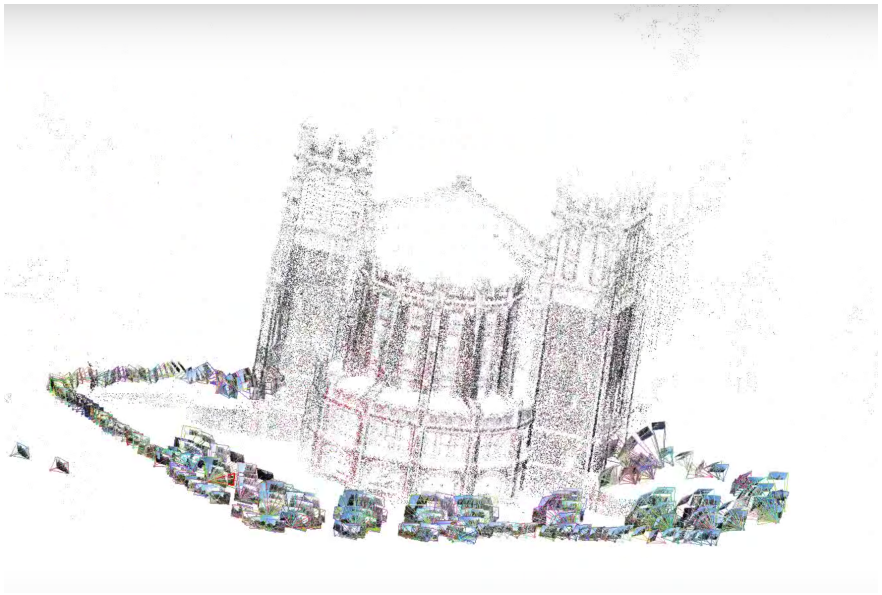
Assume we obtain an image, we first extract the SIFT feature using OpenCV SIFT extractor [6]. We utilize the SIFT detector in current OpenCV non-free module for Android, since Glass is using an Android system.

Then all the features of the new image are matched with the features in the SfM model. Since there are hundreds of thousands of features in even a small model, a sequential matching is time consuming and impractical. We apply the Fast Library for Appropriate Nearest

[6] http://docs.opencv.org/2.4/modules/nonfree/doc/nonfree.html



(a)



(b)

Figure 5.3: (a) An indoor SfM model example. (b) An outdoor SfM model example

Neighbor (FLANN) to speed up the procedure [67]. Once sufficient feature matching is achieved, Perspective-n-Points (pNp) algorithm [53] is used to calculate the camera's pose.

(1) SIFT feature extraction with OpenCV library

Different from the SIFT extraction in the SfM model building stage, where there are a larger number of images to process, here only one or a few images are needed to extract feature from. Standalone OpenCV SIFT detector without GPU acceleration is sufficient for task like this in terms of time requirement.

(2) Feature matching using FLANN

To match a feature from a new image against a feature database of a model, an intuitive approach is to use the brute-force method. The brute-force method takes the descriptor of one feature in the image feature set and matches it with all other features in the database feature set using some distance calculation (e.g. Euclidean distance). The feature with the smallest distance is returned and labeled as the corresponding matched feature.

However, when an environment feature set is becoming larger, brute-force turns to be time-consuming and other methods are needed to increase the matching speed—FLANN-based matching is one of them. FLANN contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features.

(3)Pose solver using PnP algorithm

Estimating the camera pose from n 3D-to-2D point correspondences, or pose solving, is a fundamental and well understood problem in computer vision field. The general version of the problem requires estimating the six degrees of freedom extrinsic matrix (rotation and translation) and the intrinsic parameters (including focal length, principal point, aspect ratio and skew). It could be solved with a minimum of 6 correspondences, using the well-known Direct Least-Squares (DLS) algorithm [28]. However, when some of the camera parameters such as intrinsic parameters are known, this problem can be simplified to be Perspective-n-Point (PnP) problem [53]. The procedure can be formalized in the Equation 5.2.

$$sP_c = K[R \quad t]p_w \tag{5.2}$$

where $p_w = [x, y, z, 1]^T$ is the homogeneous world point, $p_c = [u, v, 1]^T$ is the corresponding homogeneous image point, K is the matrix of intrinsic camera parameters, s is a scale factor for the image point, and R and T are the desired 3D rotation and 3D translation of the camera (extrinsic parameters) that are being calculated.

In our application, since all the test images are from the camera on the Google Glass, we carry out a calibration procedure and calculate the camera intrinsic matrix K using Zhang's calibration method [109]. Therefore the calculated matrix K can be used as the known parameters for utilizing the PnP algorithm for solving the extrinsic parameters. The user's location is represented in the translation vector T. The user's orientation is described in the rotational matrix R.

The above process assumes that the features are perfected detected and matched. However, for images from nature environment, feature mismatching or matching error is unavoidable and can lead to false or inaccurate pose estimation. To improve the localization robustness and remove matching outliers, RANdom SAmple Consensus (RANSAC) [22] is utilized. In our implementation, solvePnPRansac from OpenCV is used for the extrinsic matrix estimation [7].

5.1.3 Experiment results

We carried out our experiment with the campus indoor model as shown in Fig. 5.2a. Here are some statistics about our environment and its model. In total there are 256 images for creating the SfM model. There are 706,756 matched SIFT features pairs, and there are 86,177 3D points reconstructed. The physical room size is around $6m * 4m * 3m$.

Each 3D point has at least two associated SIFT features, but not every 3D point has the same amount of features. For some physical locations, if the environment has rich texture and therefore the features are very dense, each 3D point maps to high amount of 2D SIFT

[7] http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html#solvepnpransac

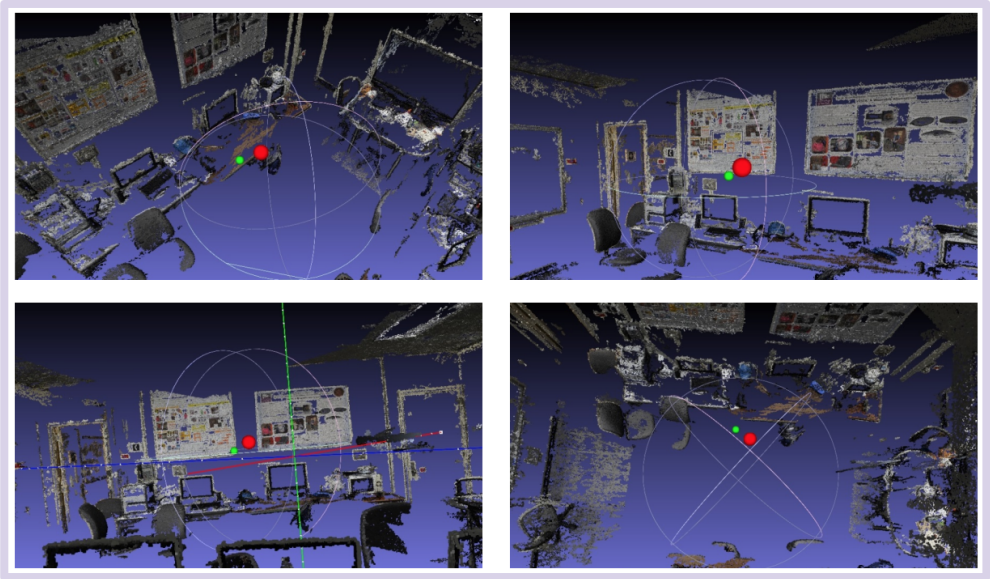


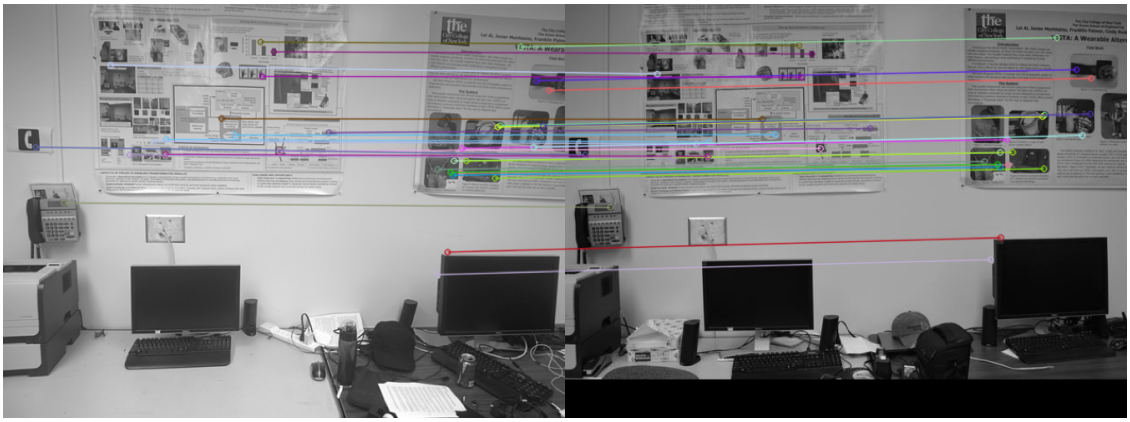
Figure 5.4: A sample localization result

features, while for some others, the features are sparse and each 3D point has fewer associated SIFT features. On average, each 3D point has around 8.2 related features.

Fig. 5.5a shows a sample SIFT matching result. The left image is a new image captured with Google Glass at a different time after the SfM model is built. The right image is a sample image captured using a Nikon D3200 camera. The lines with different colors are some SIFT matching result. Fig. 5.5b shows some sample SIFT matching results using one image from the Glass and the other images in the database.

After we find a matching descriptor for a given image feature, we can now find its 3D point, since all the SIFT feature in the database are related with a 3D point in the model. We can then utilize these 3D points along with the features 2D coordinates to find the cameras position and orientation.

Fig. 5.4 shows a localization example, where the red dot stands for the location, and the green dot stands for the heading direction. In this example, the ground truth location is (0.61, 0.30) in meters with a 45-degree heading direction, and the estimated location is (0.49, 0.28) in meters with an estimated heading of 40.2 degrees. Since it is for navigation purpose within a floor plan, we only concern about the position on the plan, and does not



(a) Image matching result example using a Glass image and a database image

(b) SIFT matching result to multiple database images

Figure 5.5: A sample SIFT matching result

concern about the height of the camera. The same thing for the orientation—we reduce the rotation matrix to a 2D case, where we only care the heading direction on the floor plan. In this example, the error in distance is 0.12 meters and in the heading angle is 4.8 degrees.

Figure 5.6: Distribution of the testing locations

For further evaluating the accuracy and robustness of the system, we conduct an extensive experiment by testing our algorithm from 13 different locations of the lab, the distribution of the locations is shown in Fig. 5.6. Multiple images, ranging from 1 to 3, heading to different orientations are taken at each location. The arrows in Fig. 5.6 illustrates the orientation when each testing image was captured.

We build a global coordinate system in the lab and manually label each testing image's location and orientation as the ground truth, as shown as round dot in Fig. 5.7. The localization results are calculated with our proposed algorithms and are visualized in the Fig. 5.7 as crossing marks. We can see that majority of the results are within the circle of one grid around the ground truth, which corresponds to 30cm in physical space.

The accuracy of the location results is shown in Fig. 5.8, with the average of 19cm, which is sufficient with most of the assistive localization tasks.

Note that there are three images that have very large errors (image No. 15, 17 and 23). This is because the physical scenes covered by these images have very spare texture, with

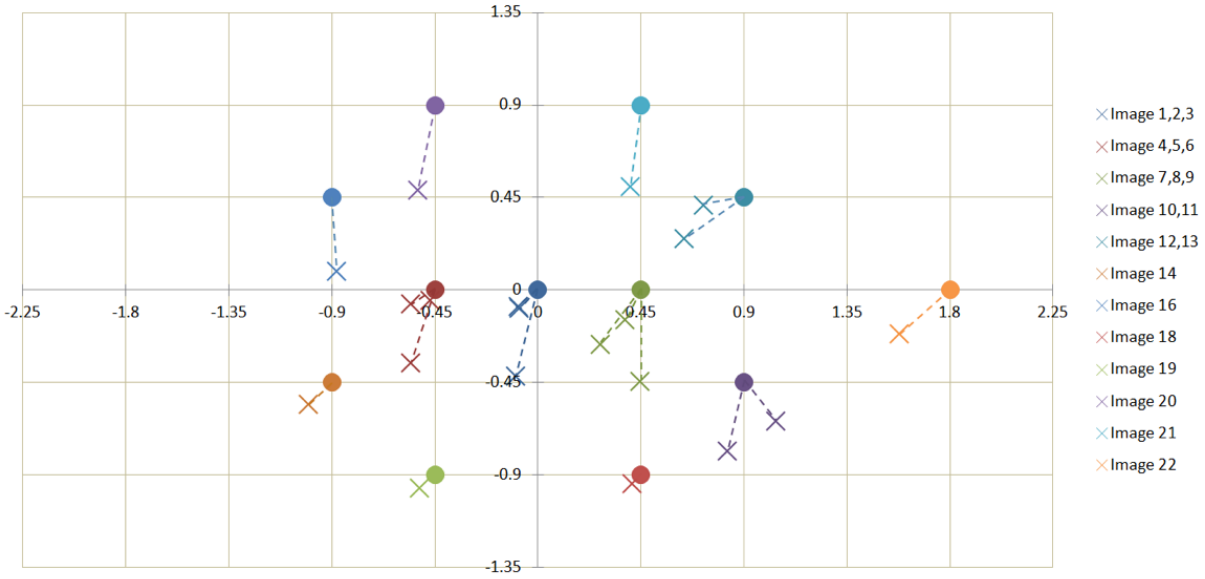


Figure 5.7: Ground truths (dots) and estimated results (crossing) of the testing images

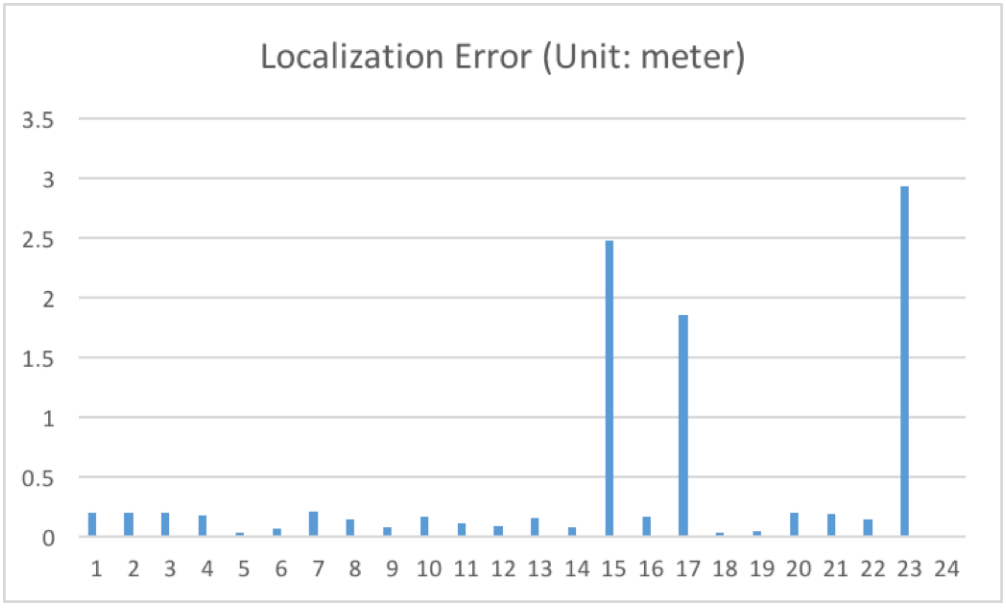


Figure 5.8: Estimated results statistics

objects such as white walls, black board or TV screen. Therefore very few SIFT features can be extracted from them, or the detected features are not distinguishable. These outliers shall not be counted to the final estimation and can be eliminated by setting a threshold for the number of effective SIFT features.

5.2 Multi-omni-view based refinement

Ideally the 2D omnidirectional image based indexing approach should find the exact location of an input frame if the corresponding location was modeled in the omnidirectional database. However, because only a limited number of discrete locations are modeled, especially in the single path modeling approach, and the existence of noises, the resulting location for the input query image may not be as accurate as required. Further approach shall be utilized to refine the localization result.

Instead of using a different camera to obtain a 3D model and then to refine the camera's 3D location, in this section, we want to use the omnidirectional images directly, without a 3D model. Observing that if we know two or more database omnidirectional images' locations, and a new query omnidirectional image's relationship with these known images, we can utilize the geometry relationship among them to estimate the query image's relative 3D pose (i.e., moving direction and heading angles) for further accuracy improvement.

Section 5.2.1 will introduce the basic problem context and mathematical modeling of the problem. Section 5.2.2 will show our experiment on how to utilize the existing omnidirectional images' location and relationship with the query image for location refinement.

5.2.1 Geometric constraints based localization

We assume that after the GPU-based parallel aggregation algorithm discussed in Chapter 4, we find the most similar omnidirectional image in the database which was modeled and mapped to the physical space beforehand, using for example floor plan. Because of the modeling accuracy and environment noise, these two images have very similar visual information, meaning we can find corresponding matching features between them, but they are not in the same location. In this subsection, we will show how to find the query omnidirectional image's relative location against the matched omnidirectional image in the database whose position was labeled in the modeling process.

We model the omnidirectional imaging system with the spherical camera models [94][2], instead of the traditional pinhole camera model. Without losing generality, let's assume that there are two spherical camera coordinate systems—one for the query omnidirectional image whose position is unknown, and the other for the reference omnidirectional image in the database whose position is already recorded. The geometric relationship is illustrated in the Fig. 5.9, where the bottom left is a query coordinate system, and the top right is a reference coordinate system. We align the origin and the three axes of the world coordinate system with the corresponding origin and axes of the query spherical camera coordinate system. We parametrize the motion from the query spherical camera coordinate system to the reference spherical camera coordinate system with two arguments: α and β, where α stands for the angles the query coordinate system shall rotate counterclockwise to be able to align with the reference coordinate system, and β stands for which direction the query coordinate system shall translate to in order to align itself with the reference coordinate. In other words, the query spherical coordinate system can align with the reference spherical coordinate system by first a translation (represented by a translating direction angle β) and then a rotation (representing by a heading angle α).

Note the motion of the camera is a 2D motion in the floor plane, with two parameters—a rotation angle α and a heading direction β. This is because we are assuming that the users, i.e., the visually impaired people, are standing on the floor, so the z axis—the vertical direction—can be ignored.

After the problem is formulated, we can estimate the two parameters by utilizing the relationship between these two images. We detect and match local SIFT features between the two omnidirectional images. Before we move to the relationship formulation, we will need to find the geometric relationship between the pixels in the omnidirectional image $I(u_{omni}, v_{omni})$ and our unit spherical $(x_{sph}, y_{sph}, z_{sph})$ models, which is formalized as

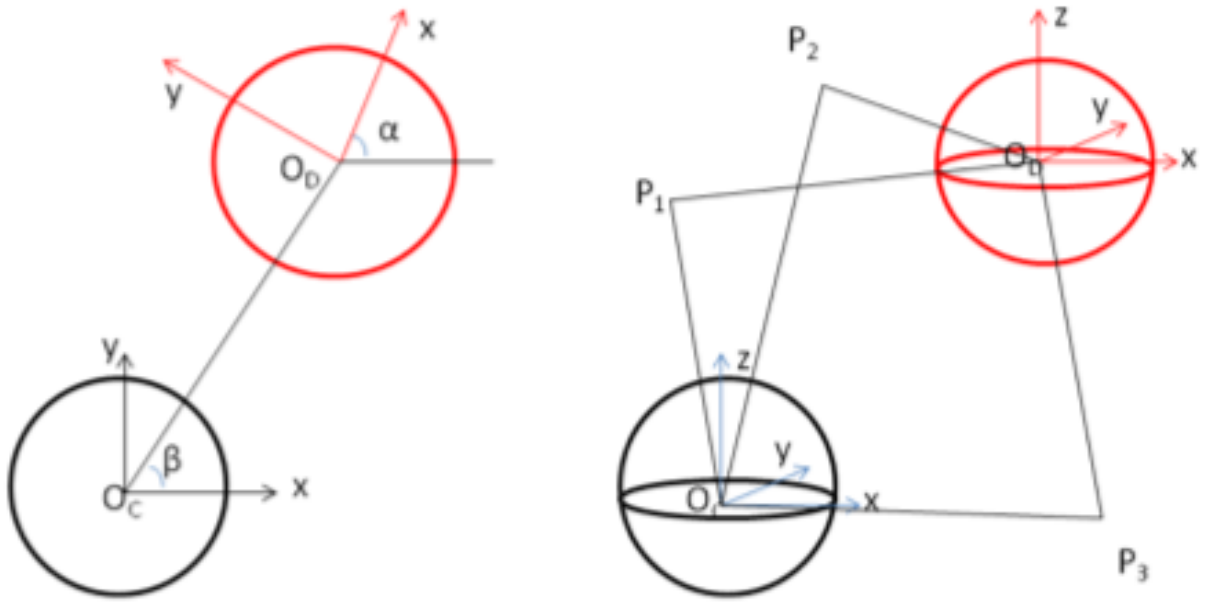


Figure 5.9: (a) Top view of current and destination locations. (b) 3D view of current and destination locations

$$\begin{cases} x_{sph} = cos((H - v_{omni}) * D) * cos(u_{omni}/W * 2\pi) \\ y_{sph} = cos((H - v_{omni}) * D) * sin(u_{omni}/W * 2\pi) \\ z_{sph} = sin((H - v_{omni}) * D) \end{cases} \tag{5.3}$$

where W is the width of the cylindrical image, H is the height (in pixel) of the center of the projection of the camera from the ground in the cylindrical representation (estimated by the calibration procedure), and D is degree per pixel in the vertical direction. With Equation 5.3, we can calculate the corresponding camera coordinates $(x_{sph}, y_{sph}, z_{sph})$ for any image point $I(u_{omni}, v_{omni})$ in the un-warped cylindrical image space.

After we detect, describe, and match SIFT features for the query omnidirectional image and the reference omnidirectional images, we can obtain corresponding image feature pairs. Given a pair of image feature points, we can use Equation 5.3 to calculate its corresponding unit sphere point pairs. These point pairs satisfy the constraints expressed in Equation 5.4

$$p_r^T \times E \times p_l = 0 \tag{5.4}$$

where p_r and p_l are the 3D spherical points before and after the motion, and E is the essential matrix between the two views [2] :

$$E = \begin{bmatrix} 0, 0, cos\alpha sin\beta - sin\alpha cos\beta \\ 0, 0, -sin\alpha sin\beta - cos\alpha cos\beta \\ -sin\beta, cos\beta, 1 \end{bmatrix} \tag{5.5}$$

Finally, using equation 5.4 we can estimate the two angles α and β by using at least three pairs of points in two images. Note that by using only two omnidirectional images, we cannot determine the translate vector amplitudes, we need to involve a third omnidirectional image to determine the scale factor [2]. The third image with known location can be easily obtained by find the second closest match of the current image in the database. Experimental results on the synthetic and real data are shown in subsection 5.2.2.

5.2.2 Experiment using multi-view omnidirectional vision

In this subsection, we will show experiment results on utilizing the synthetic data and real omnidirectional image data for localization. Since there is unavoidable noise in SIFT feature detection and matching, we will first examine how the detection/matching error will influence the parameter estimation using synthetic data. Then we will show our real data experiment as well as robustness improvement via the RANSAC algorithm [22].

(1) Error analysis on simulated data

As we can see from Equation 5.3, because of feature detection noise, there will be errors between the real 3D space position and calculated unit sphere position. This error propagates and influences the accuracy of the estimated pose parameters when we are calculating the

parameters with Equation 5.4. We will first examine how the noise in the feature detection process will influence the angle estimation accuracy.

In our experiments, we evenly distribute random noise with interval $[-n, n]$, where $n = 2, 4, 8$ pixels, which is added to the original point image coordinates. The estimated angles with and without the noise are shown in Table 5.1.

Parameters	Ground truth	S_1			S_2		
pixel error	0	2	4	8	2	4	8
α	60	61.9	61.3	64.4	59.7	57.6	61.2
β	45	47.3	46.3	50.4	45.6	43.4	48.3

Table 5.1: Error analysis results in pose estimation

Two sets of results are obtained here for estimating angle and using equation 5.4. When estimating and from the essential matrix $E = [e_{ij}]_{3\times3}$ in equation 5.5, we can determinate the scale based on e_{13} and e_{23} (i.e., the third column of E, leading to solution $S1$ in Table 5.1) or e_{31} and e_{32} (i.e., the third row of E, leading to solution $S2$). As we can see from the Table 5.1, errors in feature point localization in omnidirectional images only cause minor errors in the final estimation angles—less than 5 degrees in average even there are 8 pixels offset the ground truth feature location. Since we only need approximate moving direction instructions for visually impaired people, this amount of error is within the acceptance limits of angle errors. Solutions $S1$ and $S2$ are comparable from our simulated experimental results, but $S1$ gives slightly more stable results. So we choose Solution $S1$ for our real data experiments.

(2) Real data experiment with RANSAC

RANdom SAmple Consensus(RANSAC) [22] is an iterative method to estimate parameters of a mathematical model from a set of observed data which possibly contains outliers. Given N pairs of corresponding points, three point pairs are randomly selected for estimating a set of angle pair (α, β). Then the average error of the estimation is calculated as shown in Equation 5.6.

Figure 5.10: An example of reference omnidirectional image (top) and new query omnidirectional image (bottom)

$$D(\alpha, \beta) = \sum_{i=1}^{N} p_{i_r}^T \times E \times p_{i_l} \tag{5.6}$$

where p_{i_r} and p_{i_l} are corresponding points of the i_{th} $(i = 1, 2, ..., N)$ selected points pairs from the matched points candidates. Fig. 5.10 shows a real localization refinement example, where the top figure is the reference omnidirectional image from the modeling database whose position is known, and the bottom is a new query omnidirectional image whose rough location is provided but waiting for refinement.

Ten pairs of points are selected as the testing point pairs from these two images for RANSAC based parameter estimation. Every time, three of the ten points are used to calculate a set of pose parameters and and all the ten points are used to evaluate the result during RANSAC.

Fig. 5.11 shows the estimated results of the 120 (${C_{10}}^3$) trials. The horizontal axis stands for the trial index of the experiments, while the vertical axis stands for the estimated α and β results from the triples, legended as $AlphaC1Value$ and $BetaC1Value$. The yellow line indicates the best estimated β value of all the trials, and the pink line indicates the best

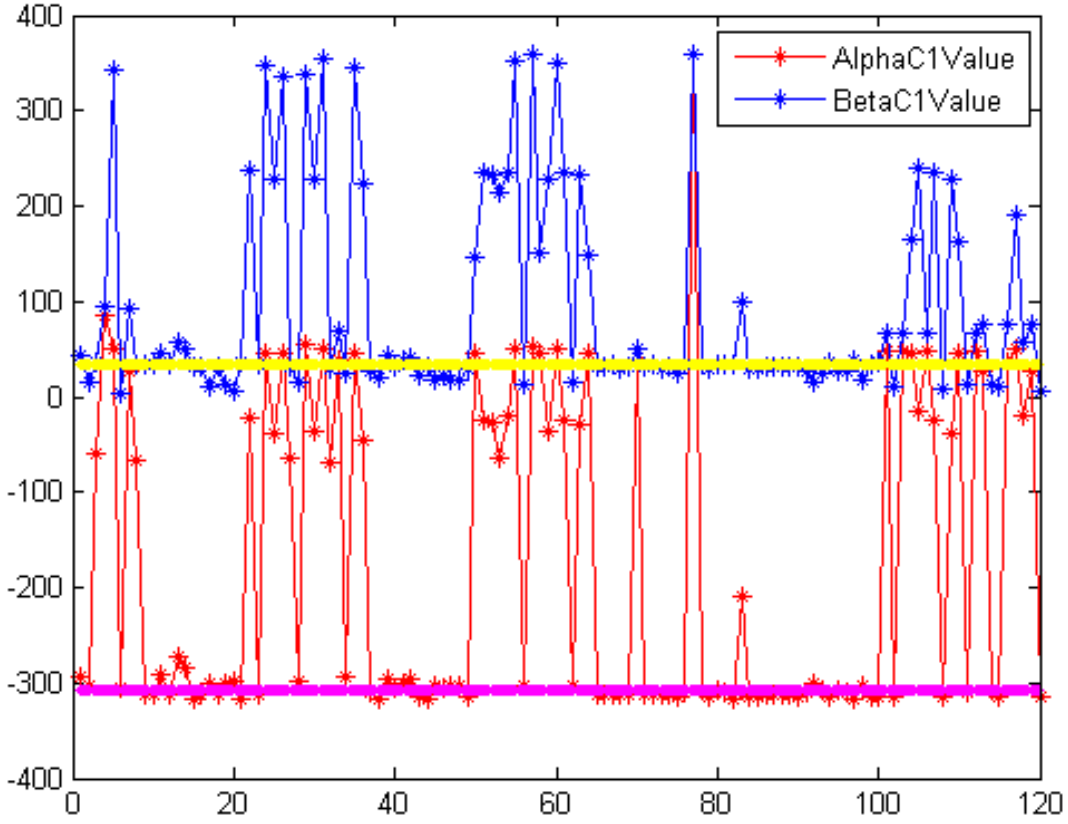


Figure 5.11: Estimated pose parameter results

estimated α value. As we can see that even though there are a few noisy results whose values are away from the final α and β result (the red and blue impulse values), by utilizing RANSAC, we can eliminate these errors and integrate the correct results. Using RANSAC can also improve the accuracy of the estimation compared with using just mean values of the 120 estimation results, as shown in Table 5.2. The errors between the estimated and the ground truth are also listed in the table, while each error value is calculated as the ratio of difference between an estimation and the ground truth, and the full range of the angles (360).

Parameters	Ground truth	Mean value and error		RANSAC value and error	
α	45	58.1	3.6%	52.1	2.0%
β	27	34.0	1.9%	32.8	1.6%

Table 5.2: Real data estimated results

5.3 Dense reconstruction based refinement

Previous work in the computer vision community uses 2D images for localization and navigation, which may be challenging when lacking of texture in indoor environments. In this section, we use fully 3D information automatically captured by a tablet with a depth sensor to refine localization results.

In order to facilitate navigation for the visually impaired, we design, implement and evaluate the system to calculate the position and orientation of the device with the following two steps: (1) use a tablet with depth sensor to pre-build a large 3D indoor environment; (2) apply Iterative Closest Point (ICP) algorithm [7] to a newly captured RGB-D (color and depth) image to calculate the users new position and orientation. The system includes three components: environmental modeling, pose estimation algorithm, and GUI design. The experiment tested with real model within a university laboratory shows a real-time and accurate performance.

5.3.1 System design and implementation

We construct the system with three components: an environmental modeling and optimization module, a pose estimation module, and a GUI module. Fig. 5.12 shows the system diagram. The environmental modeling and optimization module is to utilize the device depth information to create 3D indoor environmental model and use corresponding color information to optimize the model. In the pose estimation module, a newly captured RGB-D image is used to align itself with the pre-built 3D model for calculating the devices location. An easy-to-use GUI module is designed with voice feedback for the visually impaired users.

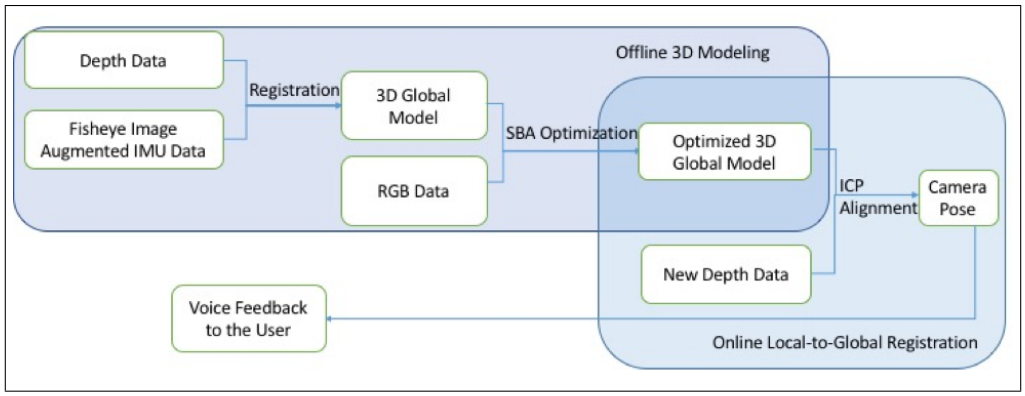


Figure 5.12: 3D-sensor-based indoor localization system diagram

(1) 3D modeling and optimization

With recent advance of depth and motion sensors (e.g. Google Tango Tablet[8], Microsoft Kinect[9] and Structure IO[10]) and development of 3D reconstruction technology [18], we can apply a dense 3D depth sensor for the localization task.

We use both 2D images and 3D depth information captured by a Google Tangle Tablet to improve robustness of location estimation. In this section, a pre-built model of the environment shall exist before a localization service is provided. The procedure of creating the model is shown as follows:

(i) Capturing piecewise local 3D models of the environment and measuring the corresponding pose information of the tablet;

(ii) Fusing the piecewise 3D models by transforming each model to a common coordination system using the given pose information of each local 3D model, and a global optimization framework is applied to increase the accuracy of fusion process [56].

Since Project Tangos Tablet device uses an Inertial Measurement Unit (IMU) for providing camera poses, augmented using built-in fisheye visual features, there are motion drifts accumulated while the sensor viewing the environment. Thus, loop closure is needed to adjust the generated global model by traversing the same area for a second time, and distribute

[8] http://get.google.com/tango/
[9] https://developer.microsoft.com/en-us/windows/kinect
[10] http://structure.io

drift errors along the motion path. In our work, we use Sparse Bundle Adjustment (SBA) [56] for the loop closure.

The general idea of SBA is first to find a sequence of pairs of 2D image features and corresponding 3D points, and then project the 3D points back to the 2D images using the known camera intrinsic parameters and unknown extrinsic parameters, which includes the cameras pose information. By minimizing the distance between the projected features and observed features, we estimate the optimized camera motion parameters.

(2) Pose estimation

As we have discussed in previous step, the major component of the system is the accurate 3D reconstruction of the global model, which has been done offline on a separate machine. In our implementation, we have a reconstructed global model generated by stitching RGB-D data of all the frames together with the camera pose data of each frame. To localize the tablet device, a user captures a new RGB-D data at a new location by holding the tablet, and the indoor navigation system then registers the new data with the pre-built global 3D model using the ICP algorithm [108]. The registration algorithm will return a transformation matrix whose rotation matrix and translation vector can give us the orientation and position of the camera, respectively.

(3) Graphic User Interface (GUI) design

In order to make the system applicable to the visually impaired, we have designed a GUI on the tablet using audio-tactile feedback, which reminds and guides users to adjust the tablet poses when capturing new RGB-D data for localization, and then after the matching is done, informs user the estimated location.

Fig. 5.13 shows an image of the GUI layout. Since blind users cannot see the screen and are usually not comfortable with complex GUI, we simplify the interface by adding a big button on the right bottom part of the screen. Once the user presses the button, the

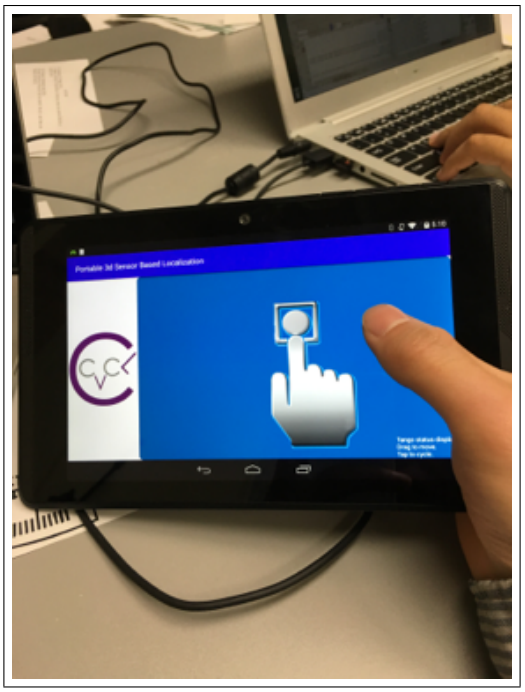


Figure 5.13: GUI design. The large right button is for starting localization service, and the rest area can be used for displaying information for administrative purpose

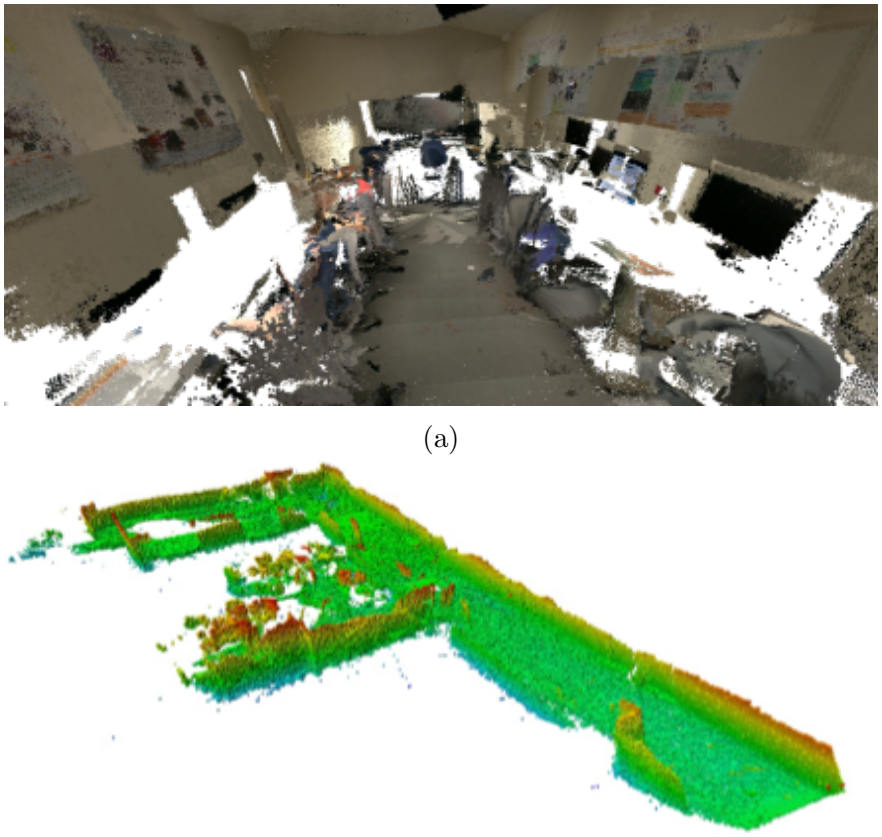
tablet will begin to capture surrounding 3D data, and feedback to the user via voice after calculating the location.

5.3.2 Experiments results

The Google Tango Tablet, which has a 3D sensor onboard, is adopted in our experiments. We have built a few large global 3D models in a campus building, as shown in Fig. 5.14a. In Fig. 5.14a, we build a 3D model for an indoor research lab by scanning the room for 5 times, each with a different tilt angle. A short video of this model can be accessed from this link[11]. In the Fig. 5.14b, half of the 8th floor of the NAC building at CCNY are scanned and visualized.

We then captured some RGB-D frames at different locations and apply the aforementioned registration method to estimate where each RGB-D frame is captured, i.e, the location of the user. The system can estimate correctly locations in the experiments.

[11] https://goo.gl/ILFlTg



(a)

(b)

Figure 5.14: A partial global 3D model including corridors and rooms

Fig. 5.15 shows some experimental results. The green part represents the partial global 3D model of Fig. 5.14a, whereas the small red model (a corner of a lab) indicates a local 3D model, which is a RGB-D frame captured after the global 3D model is built. Fig 5.15 shows that newly captured frame is correctly aligned with the global model, which indicates the location is correctly estimated. A short video of this process can be accessed via the link[12].

An optimized 3D model eliminating drifts is critical for an accurate indoor localization. In our work, we utilize the Sparse Bundle Adjustment (SBA) [56] for improving the 3D model built with Tango device. We first extract SIFT [57] features on all color images corresponding to each depth image, and then match them pairwise. After that, we select the qualified SIFT features by setting a threshold to make sure each qualified feature appears at

[12] https://goo.gl/VqBZrp

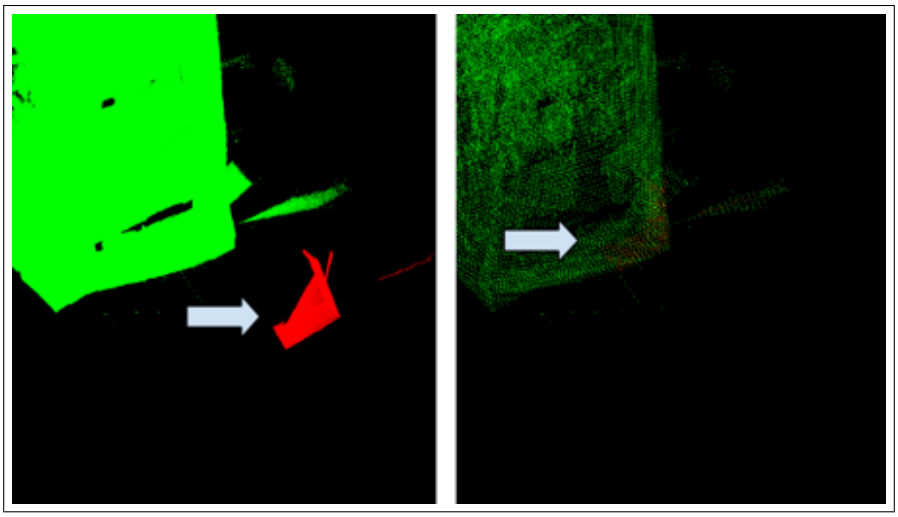
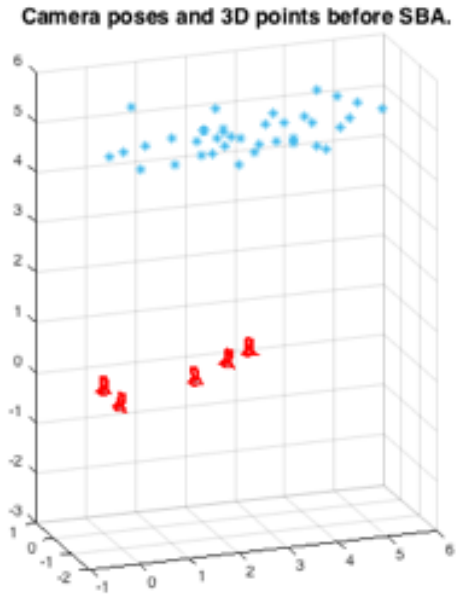
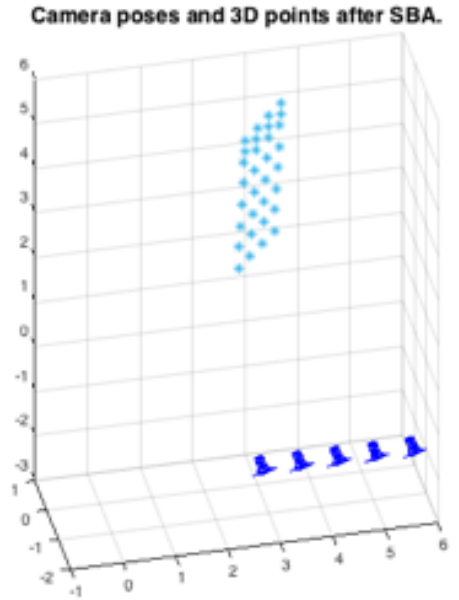


Figure 5.15: Applying the ICP based registration algorithm for localization. Left is a local model and a partial global model (in green) before registration, while right figure is the registration result

least multiple images. Then we find the corresponding 3D points for each feature from the depth images. The SBA then accepts the features and corresponding 3D points and outputs optimized camera poses. Fig. 5.16 shows SBA optimization on the synthetic data, where Fig. 5.16(a) shows the camera poses and 3D points before SBA optimization, and Fig. 5.16 (b) shows the new poses and points after the optimization.



(a)



(b)

Figure 5.16: Camera poses and 3D points before and after SBA optimization

Chapter 6

Conclusion and Discussion

This thesis presents a vision based assistive indoor localization approach with various techniques in different stages and from different perspectives for helping visually impaired people to localize in and navigate through indoor environments. Different from many other computer vision research, whose research problems are already well defined and formalized by the community, and whose major tasks are to apply their developed algorithms on standard datasets by tuning the parameter of models and evaluate the performance, this work studies the navigation need of visually impaired people, and then develop techniques in data collection, model building, localization, and user interfaces in both pre-journey planning and real-time assistance.

As a summary, we use a smart phone with a panoramic camera, and a high performance server architecture to ensure the portability and mobility of the user part and take advantage of the huge storage as well as the high computation power of the server part. An image indexing mechanism is used in finding the location of an input image (or a short sequences of images/multiple images). To improve the query speed and ensure a real-time performance, we use many-core GPUs to parallelize the query procedure.

For ensuring automatic environmental modeling using the previous omnidirectional imaging system, we utilize the building AutoCAD files for traversability checking beforehand for

finding the optimal necessary paths in navigation from point A to point B in an indoor environment. A coarse-to-refine approach is used to obtain an initial localization efficiently and then to further improve the localization accuracy from three perspectives: (1) we use Structure-from-Motion (SfM) mechanism to create a 3D environment model, and use PnP algorithm to localize a new image within the model. (2) Using pre-calibrated omnidirectional images and new input omnidirectional image, as well as their geometric relationship for refine the new image's location and the orientation. (3) Using dense 3D scanner sensor, for example, Project Tango sensor, to obtain local 3D model, and then matching itself with the pre-built 3D global model for calculating the device's accurate location and orientation. While the refinement work is still in its early stage, it holds promising with the development of the hardware, especially the trend that visual information acquisition device that are becoming more light, portable and integrated. For example smart phones and wearable glasses are available to general public, and 3D scanner features are integrating into the smart devices themselves, such as iPhone 7 plus and Project Tango phone. We foresee that vision based indoor localization, and its potential applications such as assistive navigation, will attract more researches and widely integrating into people's daily life.

Candidate's Publications

Book Chapters, Journals and Conferences

1. **Feng Hu**, Hao Tang, Zhigang Zhu, "Computer Vision for Assistive Indoor Localization", *Book Chapter in Assistive Computer Vision*, Eds. Marco Leo and Giovanni Maria Farinella, the Computer Vision and Pattern Recognition Series, Elsevier, to appear in 2017 (invited)

2. Wei Sun, Ning Duan, Peng Ji, Renjie Yao, Chunyang Ma, Jinchang Huang, **Feng Hu**. "Intelligent In-Vehicle Air Quality Management: A Smart Mobility Application Dealing With Air Pollution In The Traffic". *23rd World Congress on Intelligent Transportation Systems*, Melbourne, Australia, October, 2016.

3. **Feng Hu**, Ning Duan, Jingchang Huang, Wei Sun, Zhigang Zhu. "A Framework of Mining Personalized Climate Preference for Self-Driving Vehicles". (drafted), New York, NY, U.S.A, September, 2016

4. Hao Tang, Norbu Tsering, **Feng Hu**, "Automatic Pre-Journey Indoor Map Generation Using AutoCAD Floor Plan", *Journal on Technology and Persons with Disabilities*, Vol.4, page 176-191, Nov. 2016

5. **Feng Hu**, Norbu Tsering, Hao Tang, Zhigang Zhu, "Indoor Localization for the Visually Impaired Using a 3D Sensor", *Journal on Technology and Persons with Disabilities*, Vol.4, page 192-203, Nov. 2016

6. **Feng Hu**, Zhigang Zhu, Jianting Zhang, "GPU-accelerated Hierarchical Panoramic Image Feature Retrieval for Indoor Localization", *Technical Report*, New York, NY, U.S.A, Jun. 2016

7. Hao Tang, Norbu Tsering, **Feng Hu**, "Automatic Pre-Journey Indoor Map Generation Using AutoCAD Floor Plan", *31st Annual International Technology and Persons with Disabilities Conference*, San Diego, CA, U.S.A, Mar. 2016

8. **Feng Hu**, Norbu Tsering, Hao Tang, Zhigang Zhu, "RGB-D Sensor Based Indoor Localization for the Visually Impaired", *31st Annual International Technology and Persons with Disabilities Conference*, San Diego, CA, U.S.A, Mar. 2016

9. **Feng Hu**, Zhigang Zhu, Jeury Mejia, Hao Tang, Jianting Zhang, "Real-time indoor assistive localization with mobile omnidirectional vision and cloud GPU acceleration", *Journal of Real-Time Image Processing (pending)*, Oct., 2015

10. **Feng Hu**, Zhigang Zhu, Jianting Zhang, "Mobile Panoramic Vision for Assisting Blind via Indexing and Localization", *2014 European Conference on Computer Vision(ECCV'14) Workshops*, Zurich, Sep. 6-12, 2014

11. Tuotuo Li, **Feng Hu**, Zheng Geng, "Structured-light 3D Surface Imaging Technology". *Journal of Network New Media*, Vol.1, No.1, Jan. 2012

12. **Feng Hu**, Tuotuo Li, Jason Geng, "Constraints-Based Graph Embedding Optimal Surveillance-Video Mosaicing", *First Asian Conference on Pattern Recognition(ACPR'11)*, Beijing, China, Nov., 2011 (Oral Presentation)

13. Tuotuo Li, **Feng Hu**, Jason Geng, "Geometric Calibration of a Camera-Projector 3D Imaging System", *The 10th International Conference on Virtual Reality Continuum and Its Applications in Industry*, Hong Kong, China, Dec. 2011

14. Tuotuo Li, **Feng Hu**, Zheng Geng, "Real-Time Dense 3D Reconstruction with Active Vision System via Graphic Hardware Acceleration", *International Conference on Opto-Electronics Engineering and Information Science*, Xi'an, China, Dec., 2011

Posters and Demos

1. **Feng Hu**, Kenichi Yamamoto, Zhigang Zhu, "3D Assistive Indoor Localization with a Google Glass", *CIE-USA/GNYC 2016 Annual Convention*, New York, U.S.A, Oct. 2016.

2. Jeury Mejia, **Feng Hu**, Hao Tang, Zhigang Zhu, "iPhone Indoor Navigation System for Visually Impaired People", *2016 Emerging Researchers National (ERN) Conference in STEM*, Washington, D.C, U.S.A, Feb. 2016

3. Norbu Tsering, **Feng Hu**, Hao Tang, "RGB-D Sensor Based Indoor Localization for Visually Impaired", *2016 Emerging Researchers National (ERN) Conference in STEM*, Washington, D.C, U.S.A, Feb. 2016

4. Kenichi Yamamoto, Feng Hu, Zhigang Zhu, "Assistive 3D Indoor Localization With a Wearable Glass", *2016 Emerging Researchers National (ERN) Conference in STEM*, Washington, D.C, U.S.A, Feb. 2016

5. Jeury Mejia, **Feng Hu**, Hao Tang, Zhigang Zhu, "Assistive Indoor Navigation System for Visually Impaired People", *2015 Emerging Researchers National (ERN) Conference in STEM*, Washington, D.C, U.S.A, Feb. 2015

6. **Feng Hu**, Jianting Zhang, Zhigang Zhu, "A Smartphone-based Navigation System for Visually-impaired People", *7th Annual Graduate Student Symposium(GSS'14)*, City College of New York, New York, U.S.A, Apr., 2014 (Third Prize Winner)

7. **Feng Hu**, Zhigang Zhu, "Where am I? -Indoor Localization and Navigation for the Visually-Impaired People", *2nd Annual Computer Science Student Workshop(CSSW'14)*, New York, U.S.A, Mar., 2014

8. **Feng Hu**, Zhigang Zhu, "Vertical Line Detection and Matching for an iPhone Navigation System with a Portable Omnidirectional Lens", *3rd GNY Area Multimedia and Vision Meeting(GNYMV'13)*, New York, U.S.A, Jun., 2013

Bibliography

[1] Hani Altwaijry, Mohammad Moghimi, and Serge Belongie. Recognizing locations with google glass: A case study. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, Steamboat Springs, Colorado, March 2014.

[2] Mohamed Aly and J-Y Bouguet. Street view goes indoors: Automatic pose estimation from uncalibrated unordered spherical panoramas. In *Applications of Computer Vision (WACV), 2012 IEEE Workshop on*, pages 1–8. IEEE, 2012.

[3] Ali Ismail Awad and Mahmoud Hassaballah. *Image Feature Detectors and Descriptors: Foundations and Applications*, volume 630. Springer, 2016.

[4] Hernán Badino, Daniel Huber, and Takeo Kanade. Real-time topometric localization. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1635–1642. IEEE, 2012.

[5] Mortaza S Bargh and Robert de Groote. Indoor localization based on response rate of bluetooth inquiries. In *Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments*, pages 49–54. ACM, 2008.

[6] Balaguer Benjamin, Gorkem Erinc, and Stefano Carpin. Real-time wifi localization of heterogeneous robot teams using an online random forest. *Autonomous Robots*, 39(2):155–167, 2015.

[7] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.

[8] Joydeep Biswas and Manuela M Veloso. Wifi localization and navigation for autonomous indoor mobile robots. 2010.

[9] Jiuwen Cao, Tao Chen, and Jiayuan Fan. Landmark recognition with compact bow histogram and ensemble elm. *Multimedia Tools and Applications*, pages 1–19, 2015.

[10] Jonathan L Carrivick, Mark W Smith, and Duncan J Quincey. *Structure from Motion in the Geosciences*. John Wiley & Sons, 2016.

[11] Alexandre Chapoulie, Patrick Rives, and David Filliat. Appearance-based segmentation of indoors/outdoors sequences of spherical views. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1946–1951. IEEE, 2013.

[12] Sakmongkon Chumkamon, Peranitti Tuvaphanthaphiphat, and Phongsak Keeratiwintakorn. A blind navigation system using rfid for indoor environments. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2008. ECTI-CON 2008. 5th International Conference on*, volume 2, pages 765–768. IEEE, 2008.

[13] Grazia Cicirelli, Annalisa Milella, and Donato Di Paola. Rfid tag localization by using adaptive neuro-fuzzy inference for mobile robot applications. *Industrial Robot: An International Journal*, 39(4):340–348, 2012.

[14] Nico Cornelis and Luc Van Gool. Fast scale invariant feature detection and matching on programmable graphics hardware. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2008.

[15] Mark Cummins and Paul Newman. Appearance-only slam at large scale with fab-map 2.0. *The International Journal of Robotics Research*, 30(9):1100–1123, 2011.

[16] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[17] Marius Drulea, Istvan Szakats, Andrei Vatavu, and Sergiu Nedevschi. Omnidirectional stereo vision using fisheye lenses. In *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on*, pages 251–258. IEEE, 2014.

[18] Ivan Dryanovski, William Morris, and Jizhong Xiao. Multi-volume occupancy grids: An efficient probabilistic 3d mapping model for micro aerial vehicles. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1553–1559. IEEE, 2010.

[19] G.M. Farinella and S. Battiato. Scene classification in compressed and constrained domain. *Computer Vision, IET*, 5(5):320–334, September 2011.

[20] G.M. Farinella, D. Rav, V. Tomaselli, M. Guarnera, and S. Battiato. Representing scenes for real-time context classification on mobile devices. *Pattern Recognition*, 48(4):1086 – 1100, 2015.

[21] Silke Feldmann, Kyandoghere Kyamakya, Ana Zapater, and Zighuo Lue. An indoor bluetooth-based positioning system: Concept, implementation and experimental evaluation. In *International Conference on Wireless Networks*, pages 109–113, 2003.

[22] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[23] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, 43(1):55–81, 2015.

[24] Zheng Jason Geng. Method and apparatus for omnidirectional imaging, October 16 2001. US Patent 6,304,285.

[25] GoPano. Gopano micro camera adapter, April 2015.

[26] JR Guerrieri, MH Francis, PF Wilson, T Kos, LE Miller, NP Bryner, DW Stroup, and L Klein-Berndt. Rfid-assisted indoor localization and communication for first responders. In *2006 First European Conference on Antennas and Propagation*, pages 1–6. IEEE, 2006.

[27] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[28] Joel A Hesch and Stergios I Roumeliotis. A direct least-squares (dls) method for pnp. In *2011 International Conference on Computer Vision*, pages 383–390. IEEE, 2011.

[29] Rob Hess. An open-source siftlibrary. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1493–1496. ACM, 2010.

[30] Feng Hu, Tuotuo Li, and Zheng Geng. Constraints-based graph embedding optimal surveillance-video mosaicing. In *The First Asian Conference on Pattern Recognition*, pages 311–315. IEEE, 2011.

[31] Feng Hu, Norbu Tsering, Hao Tang, and Zhigang Zhu. Indoor localization for the visually impaired using a 3d sensor. *Journal on Technology and Persons with Disabilities*, 4:192–203, 2016.

[32] Feng Hu, Norbu Tsering, Hao Tang, and Zhigang Zhu. Indoor localization for the visually impaired using a 3d sensor. *Journal on Technology Persons with Disabilities*, 4:192–203, Nov, 2016.

[33] Feng Hu, Norbu Tsering, Hao Tang, and Zhigang Zhu. Rgb-d sensor based indoor localization for the visually impaired. In *31st Annual International Technology and Persons with Disabilities Conference*, San Diego, CA, U.S.A, Mar. 2016.

[34] Feng Hu, Kenichi Yamamoto, and Zhigang Zhu. 3d assistive indoor localization with a google glass. In *CIE-USA/GNYC 2016 Annual Convention*, page 50, 2016.

[35] Feng Hu, Zhigang Zhu, Jeury Mejia, Hao Tang, and Jianting Zhang. Real-time indoor assistive localization with mobile omnidirectional vision and cloud gpu acceleration. *Journal of Real-Time Image Processing(pending)*, 2016.

[36] Feng Hu, Zhigang Zhu, and Jianting Zhang. Mobile panoramic vision for assisting the blind via indexing and localization. In *Computer Vision-ECCV 2014 Workshops*, pages 600–614. Springer, 2014.

[37] Zhou Hui, Luo Fei, and Li Hui-juan. Study on fisheye image correction based on cylinder model. *Computer Applications*, 28(10):2664–2666, 2008.

[38] Ryosuke Ichikari, Tenshi Yanagimachi, and Takeshi Kurata. Augmented reality tactile map with hand gesture recognition. In *International Conference on Computers Helping People with Special Needs*, pages 123–130. Springer, 2016.

[39] Cheng Jin, Zheming Wang, Tianhao Zhang, Qinen Zhu, and Yuejie Zhang. A novel visual-region-descriptor-based approach to sketch-based image retrieval. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 267–274. ACM, 2015.

[40] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[41] Sing Bing Kang. Catadioptric self-calibration. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 201–207. IEEE, 2000.

[42] Wai Lun Khoo and Zhigang Zhu. Multimodal and alternative perception for the visually impaired: a survey. *Journal of Assistive Technologies*, 10(1):11–26, 2016.

[43] Bryan Klingner, David Martin, and James Roseborough. Street view motion-from-structure-from-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 953–960, 2013.

[44] Adriana Kovashka, Devi Parikh, and Kristen Grauman. Whittlesearch: Image search with relative attribute feedback. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2973–2980. IEEE, 2012.

[45] Karl Kraus. *Photogrammetry*, volume 1. Ferdinand Dummlers Verlag, 1993.

[46] Vladimir Kulyukin, Chaitanya Gharpure, John Nicholson, and Sachin Pavithran. Rfid in robot-assisted indoor navigation for the visually impaired. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 2, pages 1979–1984. IEEE, 2004.

[47] Hideyuki Kume, Arne Suppé, and Takeo Kanade. Vehicle localization along a previously driven route using image database. In *MVA*, pages 177–180. Citeseer, 2013.

[48] Hideyuki Kume, Arne Suppé, and Takeo Kanade. Vehicle localization along a previously driven route using an image database. IAPR International Conference on Machine Vision Applications. Kyoto, JAPAN. 2013.

[49] Kiriakos N Kutulakos and James R Vallino. Calibration-free augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):1–20, 1998.

[50] Dik Lun Lee and Qiuxia Chen. A model-based wifi localization method. In *Proceedings of the 2nd international conference on Scalable information systems*, page 40. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[51] Young Hoon Lee and Gerard Medioni. A rgb-d camera based navigation for the visually impaired. In *RSS 2011 Workshop on RGB-D Cameras*, pages 1–6, 2010.

[52] Young Hoon Lee and Gerard Medioni. Wearable rgbd indoor navigation system for the blind. In *Workshop on Assistive Computer Vision and Robotics, ECCV2014*. IEEE, 2014.

[53] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155–166, 2009.

[54] Ying Li. *Video Representation*, pages 3296–3303. Springer US, Boston, MA, 2009.

[55] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1067–1080, 2007.

[56] Manolis IA Lourakis and Antonis A Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software (TOMS)*, 36(1):2, 2009.

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

[58] Wai Lun Khoo, Joey Knapp, Franklin Palmer, Tony Ro, and Zhigang Zhu. Designing and testing wearable range-vibrotactile devices. *Journal of Assistive Technologies*, 7(2):102–117, 2013.

[59] Roberto Manduchi. *Mobile vision as assistive technology for the blind: An experimental study*. The 13th Int. Conf. on Computers Helping People of Special Needs (ICCHP 2012), Springer, 2012.

[60] Kevin Matzen and Noah Snavely. Scene chronology. In *European Conference on Computer Vision*, pages 615–630. Springer, 2014.

[61] Rainer Mautz and Sebastian Tilch. Survey of optical indoor positioning systems. In *International Conference on Indoor Positioning and Indor Navigation*, pages 1–7, 2011.

[62] Andrew Mehnert and Paul Jackway. An improved seeded region growing algorithm. *Pattern Recognition Letters*, 18(10):1065–1071, 1997.

[63] Dmytro Mishkin, Michal Perdoch, and Jiri Matas. Place recognition with wxbs retrieval. In *CVPR 2015 Workshop on Visual Place Recognition in Changing Environments*, 2015.

[64] Edgardo Molina and Zhigang Zhu. Visual noun navigation framework for the blind. *Journal of Assistive Technologies*, 7(2):118–130, 2013.

[65] Edgardo Molina, Zhigang Zhu, and Yingli Tian. *Visual Nouns for Indoor/Outdoor Navigation*, volume 7383 of *13th International Conference, ICCHP 2012*. 2012.

[66] Jorge J Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer, 1978.

[67] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.

[68] Ana Cristina Murillo, Gautam Singh, Jana Kosecka, and José Jesús Guerrero. Localization in urban environments using a panoramic gist descriptor. *Robotics, IEEE Transactions on*, 29(1):146–160, 2013.

[69] Shree K Nayar. Catadioptric omnidirectional camera. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 482–488. IEEE, 1997.

[70] Shree K Nayar. Omnidirectional imaging apparatus, June 2 1998. US Patent 5,760,826.

[71] Akio Nishimura, Tetsuro Okuyama, and Yasushi Yagi. Omnidirectional imaging system, January 26 2016. US Patent 9,244,258.

[72] Snavely Noah. Bundler: Structure from motion (sfm) for unordered image collections. http://www.cs.cornell.edu/~snavely/bundler/. Last visited 01/15/2016.

[73] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[74] G. Olmschenk, C. Yang, Z. Zhu, H. Tong, and W. H. Seiple. Mobile crowd assisted navigation for the visually impaired. In *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing*, pages 324–327, Aug 2015.

[75] Nektarios Paisios. *Mobile Accessibility Tools for the Visually Impaired*. PhD thesis, New York University, 2012.

[76] Helen Petrie, Valerie Johnson, Thomas Strothotte, Andreas Raab, Steffi Fritz, and Rainer Michel. Mobic: Designing a travel aid for blind and elderly people. *Journal of navigation*, 49(01):45–52, 1996.

[77] Betsy Phillips and Hongxin Zhao. Predictors of assistive technology abandonment. *Assistive technology*, 5(1):36–45, 1993.

[78] Kuemmerle Rainer, Grisetti Giorgio, Strasdat Hauke, Konolige Kurt, and Burgard Wolfram. g2o: A general framework for graph optimization. https://openslam.org/g2o.html. Last visited 01/15/2016.

[79] Jose Rivera-Rubio, Saad Idrees, Ioannis Alexiou, Lucas Hadjilucas, and Anil A Bharath. Mobile visual assistive apps: Benchmarks of vision algorithm performance. In *New Trends in Image Analysis and Processing–ICIAP 2013*, pages 30–40. Springer, 2013.

[80] Kenneth H Rosen and Kamala Krithivasan. *Discrete mathematics and its applications*, volume 6. McGraw-Hill New York, 2011.

[81] Torsten Sattler, Michal Havlena, Filip Radenovic, Konrad Schindler, and Marc Pollefeys. Hyperpoints and fine vocabularies for large-scale location recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2102–2110, 2015.

[82] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *2011 International Conference on Computer Vision*, pages 667–674. IEEE, 2011.

[83] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In *European Conference on Computer Vision*, pages 752–765. Springer, 2012.

[84] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[85] Davide Scaramuzza. *Omnidirectional vision: from calibration to robot motion estimation*. PhD thesis, Citeseer, 2008.

[86] Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Computer Vision Systems, 2006 ICVS'06. IEEE International Conference on*, pages 45–45. IEEE, 2006.

[87] Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. A toolbox for easily calibrating omnidirectional cameras. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5695–5701. IEEE, 2006.

[88] Stephen Se, David Lowe, and Jim Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 2051–2058. IEEE, 2001.

[89] Niko Sunderhauf, Sareh Shirazi, Feras Dayoub, Ben Upcroft, and Michael Milford. On the performance of convnet features for place recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 4297–4304. IEEE, 2015.

[90] Hao Tang, Norbu Tsering, and Feng Hu. Automatic pre-journey indoor map generation using autocad floor plan. *Journal on Technology and Persons with Disabilities*, 4:176–191, 2016.

[91] Hao Tang, Norbu Tsering, and Feng Hu. Automatic pre-journey indoor map generation using autocad floor plan. In *31st Annual International Technology and Persons with Disabilities Conference*, San Diego, CA, U.S.A, Mar. 2016.

[92] William B Thompson, Thomas C Henderson, Thomas L Colvin, Lisa B Dick, and Carolyn M Valiquette. Vision-based localization. In *DARPA Image Understanding Workshop*, pages 491–498, 1993.

[93] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1808–1817, 2015.

[94] Akihiko Torii, Atsushi Imiya, and Naoya Ohnishi. Two-and three-view geometry for spherical cameras. In *Proceedings of the sixth workshop on omnidirectional vision, camera networks and non-classical cameras*. Citeseer, 2005.

[95] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustmenta modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999.

[96] Andrea Vedaldi. An open implementation of the sift detector and descriptor. *UCLA CSD*, 2007.

[97] Andrea Vedaldi and Brian Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1469–1472. ACM, 2010.

[98] Thorsten Völkel and Gerhard Weber. Routecheckr: personalized multicriteria routing for mobility impaired pedestrians. In *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*, pages 185–192. ACM, 2008.

[99] Changchang Wu. VisualSFM : A visual structure from motion system. http://ccwu.me/vsfm/. Last visited 01/15/2016.

[100] Changchang Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). http://cs.unc.edu/~ccwu/siftgpu, 2007.

[101] Jiang Xiao, Zimu Zhou, Youwen Yi, and Lionel M. Ni. A survey on wireless indoor localization from the device perspective. *ACM Comput. Surv.*, 49(2):25:1–25:31, June 2016.

[102] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2695–2702. IEEE, 2012.

[103] Jizhong Xiao, Samleo L Joseph, Xiaochen Zhang, Bing Li, Xiaohai Li, and Jianwei Zhang. An assistive navigation framework for the visually impaired. *Human-Machine Systems, IEEE Transactions on*, 45(5):635–640, 2015.

[104] Simin You, Jianting Zhang, and Le Gruenwald. Large-scale spatial join query processing in cloud. In *Data Engineering Workshops (ICDEW), 2015 31st IEEE International Conference on*, pages 34–41. IEEE, 2015.

[105] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera pose voting for large-scale image-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2704–2712, 2015.

[106] Dong Zhang, Dah-Jye Lee, and Brandon Taylor. Seeing eye phone: a smart phone-based indoor localization and guidance system for the visually impaired. *Machine vision and applications*, 25(3):811–822, 2014.

[107] Jianting Zhang, Simin You, and Le Gruenwald. Parallel online spatial and temporal aggregations on multi-core cpus and many-core gpus. *Information Systems*, 44:134–154, 2014.

[108] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994.

[109] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.

[110] Zhigang Zhu, K Deepak Rajasekar, Edward M Riseman, and Allen R Hanson. Panoramic virtual stereo vision of cooperative mobile robots for localizing 3d moving objects. In *Omnidirectional Vision, 2000. Proceedings. IEEE Workshop on*, pages 29–36. IEEE, 2000.

[111] Zhigang Zhu, Shiqiang Yang, Guangyou Xu, Xueyin Lin, and Dingji Shi. Fast road classification and orientation estimation using omni-view images and neural networks. *Image Processing, IEEE Transactions on*, 7(8):1182–1197, 1998.

[112] Andrew Zisserman, Andrew Fitzgibbon, and Geoffrey Cross. Vhs to vrml: 3d graphical models from video sequences. In *Multimedia Computing and Systems, 1999. IEEE International Conference on*, volume 1, pages 51–57. IEEE, 1999.