iMoon: Using Smartphones for Image-based Indoor Navigation

Jiang Dong Yu Xiao Marius Noreikis Zhonghong Ou Antti Ylä-Jääski
 {jiang.dong, yu.xiao, marius.noreikis, zhonghong.ou, antti.yla-jaaski}@aalto.fi
 Aalto University
 Espoo, Finland

ABSTRACT

The adoption of indoor navigation for smartphones has been relatively slow in the past years, although it would be direly needed in complex indoor areas. The primary barriers for its adoption include the lack of fine-grained and up-to-date indoor maps and the potential deployment and maintenance cost. In this paper we investigate the feasibility of utilizing crowdsourced data for building a smartphone-based indoor navigation system, focusing on the technical challenges caused by the varying quality of crowdsourced data. We developed *iMoon*, an indoor navigation system based on sensor-enriched 3D models of indoor environment, and evaluated its performance via a field study in a public building covering around 1,100 m^2 .

iMoon builds 3D models of indoor environment from crowd-sourced 2D photos, and compiles a navigation mesh from the generated 3D models. Depending on the input for 3D modelling, indoor pedestrian paths may be partly missing from the output. *iMoon* solves this issue by integrating into navigation mesh the pedestrian paths recognized from crowdsourced user motion trajectories. With photo-based 3D models, *iMoon* supports image-based localization that identifies user's position and facing direction with photos, and provides visual navigation instructions that show when and where to turn. To reduce response delay while maintaining the accuracy of localization, *iMoon* partitions 3D models based on the density of 3D points, and selects partitions for image-based localization using Wi-Fi fingerprints. According to our experimental results, *iMoon* works properly, and our solution of indoor localization achieves competitive performance compared with traditional approaches: in most cases, a user can be located within an average of 3.85 seconds with a location error of less than 2 meters and a facing direction error of less than 6 degrees.

Categories and Subject Descriptors

C.2.4 [Computer-Communication Networks]: Distributed Systems

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author(s).

Copyright is held by the owner/author(s).

SenSys '15, November 1–4, 2015, Seoul, South Korea.

ACM 978-1-4503-3631-4/15/11.

DOI: <http://dx.doi.org/10.1145/2809695.2809722>.

Keywords

Indoor mapping; Indoor navigation; 3D modelling; Mobile crowdsensing

1. INTRODUCTION

Indoor navigation systems for smartphones would be direly needed in complex indoor environments such as airports, shopping malls and museums. Unfortunately, the adoption rate of indoor navigation systems is still very low, even though initial efforts into deploying them were taken several decades ago. There are multiple reasons for the slow progress. First of all, indoor navigation requires fine-grained and up-to-date indoor maps that provide sufficient information for calculating navigation routes and instructions. Although floor plans are widely available in public places like shopping malls and airports, most of them lack details, or have not been kept up to date. Therefore, they can rarely be used directly for indoor navigation. Secondly, even in the case where accurate indoor maps do exist, most systems rely on pre-scanned radio maps or pre-installed hardware [13, 17] for localization, which are expensive to put into use and cumbersome to maintain. Thus, we saw it worthwhile to investigate whether we could develop an alternative indoor navigation method that would not need pre-created indoor maps, pre-scanned radio maps or pre-installed hardware. Inspired by the success of mobile crowdsensing applications [24] such as Waze social GPS maps & traffic¹ and OpenSignal², we came to the conclusion that mobile crowdsensing might be the key for creating such a system. The indoor navigation system *iMoon* was born out of this investigative work.

iMoon is based on sensor-enriched 3D models that are created and constantly maintained using crowdsourced photos and sensor data collected from smartphones. The system can be bootstrapped in a new location by taking photos and collecting sensor data in random spots in the location. Furthermore, photos publicly available from the Internet can also be used. Our previous work [8] demonstrates that it is feasible to build initial 3D models for places of interests, e.g., Rock Church in Helsinki, for indoor localization purpose, by using photos downloaded from Flickr. As *iMoon* users³ enter the location, move around, and navigate using the app, the system constantly updates the 3D models with

¹<https://www.waze.com/>

²<http://opensignal.com/>

³A demo video of *iMoon* mobile app is available on YouTube: https://youtu.be/sNvf7N_s59c

Figure 1: Overview of *iMoon*, a mobile crowdsensing-based indoor navigation system for smartphones.

new photos and sensor data collected from the users' devices. In practice, the more the system is used in a location, the more accurate it will get.

iMoon utilizes several existing techniques as building blocks. The primary contribution of *iMoon* is that it provides a whole-system solution for mobile crowdsensing-based indoor navigation, and addresses several technical challenges that are caused by the varying quality of crowdsourced data.

Firstly, *iMoon* takes crowdsourced photos as the input to build 3D models (in the form of 3D point clouds) of the indoor space of interest using Structure-from-Motion (SfM)⁴ techniques [10, 30, 32], and then creates navigation meshes for path planning based on the obstacle information extracted from the 3D models. Because the crowdsourced photos are taken using different devices at random locations, the generated 3D models may not cover every corner of the space, and may include denser points in certain areas than others. As a result, the pedestrian path information needed for calculating navigation routes may be partially missing from the navigation meshes. To tackle this problem, we propose to track user trajectories using motion sensors available on smartphones, and integrate the pedestrian paths extracted from the crowdsourced user trajectories into the navigation meshes.

Secondly, with 3D SfM models, *iMoon* provides image-based localization that identifies user's position and facing direction using the photos taken in situ. The problem with this approach is that the response time increases significantly as the sizes of the 3D models increase, because location is determined by matching the 2D features in the query photo against the 3D points contained in the 3D models. To enable fast localization, *iMoon* employs model partitioning based on the density of 3D points, and selects partitions for feature matching based on Wi-Fi fingerprinting. Ex-

perimental results demonstrate good performance for this partitioning scheme.

In this paper we present the design, implementation and thorough evaluation of *iMoon*. The contributions of this paper are summarized as follows:

1) We address several technical challenges to enable *iMoon*: we combine the pedestrian paths recognized from crowdsourced user trajectories to complement the incomplete ones extracted from crowdsourced 3D models of indoor environment; we speed up the localization process by density-based model partitioning and fingerprint-based partition selection for compute-intensive operations.

2) We develop algorithms to generate simple, yet intuitive sets of choreographed visual navigation instructions to benefit people with difficulties in reading maps.

3) We implement a prototype of *iMoon* to evaluate its feasibility and performance. Experimental results demonstrate that *iMoon* works properly and has good performance. According to our experiments carried out in a real building covering around 1,100 m^2 , using a 3D point cloud generated from 2,197 photos, in most cases, *iMoon* can locate a user within 4 seconds, with a location error of less than 2 meters and a facing direction error of less than 6 degrees.

It is worth noting that the response delay can further be shortened by offloading computation to more powerful machines. In addition, the *iMoon* client can be run on wearable devices like Google Glasses to provide better user experience, especially for people with visual impairment.

2. SYSTEM OVERVIEW

We argue that a good indoor navigation system should fulfill the following requirements: (1) it should be able to locate the users precisely and efficiently; (2) it should supply the users with flexible navigation routes and user-friendly navigation instructions. To date, when people are talking about indoor navigation, it is usually assumed that fine-grained and up-to-date indoor maps already exist for navigation purpose. In reality, however, such maps are not yet

⁴SfM itself refers to the process of simultaneously computing a 3D point cloud (the structure) representing the pictured scene and the camera pose per each image (the motion).

widely available. Furthermore, today’s navigation instructions are mostly in texts or texts-converted voice, which are problematic for people having difficulties in reading maps.

The *iMoon* indoor navigation system fulfils the aforementioned requirements. The system is based on a client/server architecture. The client serves two purposes: (1) collecting photos and sensor data (i.e., accelerometer, gyroscope and Wi-Fi fingerprints) to assist the server to build 3D models for indoor environment; (2) providing localization results and navigation instructions to the users.

Accordingly, as illustrated in Figure 1, the server provides two main functions: (1) building sensor-enriched 3D models, (2) and providing navigation services based on the 3D models. These functions are explained briefly below, while more details will be given in Section 3 and Section 4, respectively.

Building sensor-enriched 3D models. Firstly, *iMoon* builds an initial 3D model from the crowdsourced photos using SfM techniques. The 3D model will be updated constantly by newly collected photos, either from the Internet or from the *iMoon* clients. Secondly, *iMoon* detects user trajectories and adds them to the 3D model. In practice, sensor data including accelerometer and gyroscope readings are used as the base for detecting user trajectories. These trajectories are automatically calibrated by the photos taken on the way. Thirdly, the Wi-Fi fingerprints collected along the user trajectories are automatically geo-referenced and added to the 3D model to enable fast localization.

3D-model-based indoor navigation. *iMoon* combines image-based localization with fingerprint-based approach for fast localization. The 3D model is partitioned into multiple smaller models based on the density of 3D points during an offline process. A localization request includes a photo and, if available, a group of Wi-Fi fingerprints collected in situ. These Wi-Fi fingerprints are used to match with the geo-referenced Wi-Fi fingerprint database. Once coarse locations are found, the model partitions that contain these positions are selected for executing image-based localization. After the source and the destination are identified, *iMoon* calculates navigation routes based on navigation meshes using path finding algorithms. The navigation meshes are first compiled from the obstacle information extracted from the 3D models, and then modified by adding the pedestrian paths extracted from crowdsourced user trajectories. For each selected navigation route, *iMoon* creates navigation instructions using textual and visual information contained in the 3D models.

3. BUILDING SENSOR-ENRICHED 3D MODELS OF INDOOR ENVIRONMENTS

This section describes how we utilize crowdsourced data in building sensor-enriched 3D models for indoor environments. The 3D models are later used for implementing indoor navigation. In Section 3.1, we describe how we apply SfM techniques to build and update 3D models, taking crowdsourced photos as input. After that, in Section 3.2, we explain how we utilize the motion sensors available on smartphones in detecting user trajectories, and in enriching the 3D models with crowdsourced user trajectories. Additionally, we add the Wi-Fi fingerprints collected along user trajectories to the 3D models (see Section 3.3).

Figure 2: The pipeline of generating sparse 3D point clouds from 2D photos using SfM techniques.

3.1 Initializing and Updating SfM-based 3D Models

iMoon applies SfM techniques to build sparse 3D point clouds from unordered 2D photos. As illustrated in Figure 2, a typical SfM pipeline includes 3 steps: feature extraction, feature matching, and bundle adjustment. Firstly, highly distinctive and invariant features are extracted from images using algorithms like Scale-Invariant Feature Transform (SIFT) [18]. After that, the SfM pipeline tries to match features between photo pairs. Some photos are automatically dropped if they do not have enough common matches with others. Finally, the matches are input for the bundle adjustment step that jointly produces optimal estimates of camera poses and locations of 3D points.

iMoon implements the SfM pipeline based on VisualSfM [2, 32], a state-of-the-art SfM tool chain. VisualSfM integrates SiftGPU [1] and parallel bundle adjustment [33] to accelerate the feature extraction and feature matching processes. The output of the pipeline includes a sparse 3D point cloud and a list of estimated camera poses. In detail, a sparse 3D point cloud is a set of data points in a 3D coordinate system. Each 3D point is represented as a tuple consisting of the 3D coordinates (location), color, and a list (showing what images and which features in these images have been used for reconstructing the particular point). For each input image, there is a camera pose included in the output. A camera pose is described with the 3D coordinates of the camera position, focal length, and rotation.

Our previous work [8] has demonstrated an opportunity to create initial 3D models of certain indoor environments from publicly available Internet photos. As it is very likely that there exist insufficient number of photos at the beginning to cover every corner of the space and to contain all the details (e.g., obstacles), in the design of *iMoon*, we choose to constantly update models with new arrivals of photos.

We extend VisualSfM to support automatic model update as described below. First, if the new photos provide more details of an area already covered by the latest 3D models, the new photos will be registered to the latest models based on feature matching. Second, if the new photos reflect changes in indoor scenes, or cover an area that has not been included in the latest 3D models yet, these photos cannot be directly registered to the latest models. In this case, *iMoon* will first create 3D models from the new photos and then try to merge them with the existing ones. For the first case (i.e., photos reflecting changes), *iMoon* will calculate the field of view for each of the new photos, and then apply a ray tracing algorithm to detect which points from the previous models lie inside the field of view. The detected points will then

be removed from the latest 3D models. In case there are insufficient common features between models (i.e., covering different areas), these models will remain disconnected. Potential solutions to the fragmentation issue will be discussed in Section 6.

3.2 Detecting User Trajectories using Motion Sensors

The *iMoon* client collects accelerometer and gyroscope readings from smartphones as users traverse in indoor environments. Based on these readings, the *iMoon* server calculates user trajectories based on dead reckoning [15], and uses photos taken on the way for trajectory calibration. The process of automatically detecting and calibrating user trajectories is implemented through two steps.

Firstly, the *iMoon* server obtains the camera position of each photo taken on the way based on the image-based localization method, which will be described in Section 4.1. The camera positions are set to be the calibration points.

Secondly, between every two successive calibration points, the *iMoon* server computes a trace based on step count, heading offset, and stride length. Steps are counted based on accelerometer data [15], while the heading offset between two steps is calculated from gyroscope readings. The stride length is first set to its default value, e.g., 0.7 meter, and then updated according to the accumulated walking distance and step count. The position after each step is reckoned based on the previous position and the accumulated heading offset. The calculated trace will be calibrated based on the locations of the two calibration points [31].

Given a set of user trajectories, *iMoon* discovers pedestrian paths by selecting the paths that have been covered by a minimum number of trajectories. By default we set the number to 2. Each pedestrian path is represented as a sequence of points. These paths are integrated into the 3D point clouds while compiling navigation meshes. More details about compiling navigation meshes will be presented in Section 4.2.

3.3 Geo-referencing Wi-Fi Fingerprints

The *iMoon* client collects Wi-Fi fingerprints while capturing accelerometer and gyroscope data along user trajectories. Based on time stamps, these Wi-Fi fingerprints can be easily geo-referenced and attached to the 3D point clouds after the user trajectories have been automatically calibrated. The geo-referenced Wi-Fi fingerprints are stored in a database which will be used for fingerprint-based 3D model partition selection, as described in Section 4.1.

4. 3D-MODEL-BASED INDOOR NAVIGATION

After building the sensor-enriched 3D models as described in Section 3, we continue to discuss how indoor navigation can be conducted in this section. We first describe indoor localization in Section 4.1; then we present how navigation meshes can be compiled in Section 4.2; in Section 4.3, we state how we create user-friendly navigation instructions.

4.1 Indoor Localization

Given a query photo, the image-based localization is expected to return the location where the photo is taken and the direction in which the camera is facing. A straightforward method is to register the query image into the 3D point clouds based on feature matching. In theory, the feature

Figure 3: An example of density-based point cloud partitioning. Each partition includes points corresponding to features extracted from no more than 100 photos. Both the width and length of each partition are larger than 5 meters. The camera positions are marked as green dots in the figure.

Algorithm 1 Pseudo code of indoor localization algorithm implemented in *iMoon*

Input: User’s query photo: U_{photo} , Wi-Fi Fingerprints: U_{Wi-Fi}

Output: User’s location: U_{loc} , User’s facing direction: U_{dir}

- 1: //Apply k-NN to get a set of coarse locations based on Wi-Fi fingerprints
 - 2: //DB_{Wi-Fi}: Database of geo-referenced Wi-Fi fingerprints
 - 3: $L_{coarse} = \text{k-NN}(U_{Wi-Fi}, \text{DB}_{Wi-Fi})$;
 - 4: //Select model partitions based on coarse locations
 - 5: $\text{Partitions} = \text{SelectPartition}(L_{coarse})$;
 - 6: //Find photos taken near the coarse locations from the selected partitions
 - 7: $\text{Photos}_{nearby} = \text{SearchPhoto}(L_{coarse}, \text{Partitions})$;
 - 8: //Identify user’s location and facing direction based on feature matching between U_{photo} and each photo in Photos_{nearby} .
 - 9: $[U_{loc}, U_{dir}] = \text{StartSfMInParallel}(U_{photo}, \text{Partitions}, \text{Photos}_{nearby})$;
-

matching is conducted between the query image and each of the images used for building the point clouds. If enough matches⁵ can be found, the features provided by the query image will be added to the point clouds, and a camera pose of the input will be returned.

The size of a 3D point cloud, in terms of the number of 3D points, increases with the number of photos involved. It results in consuming more resources to load the point cloud into memory or to go through each point in the point cloud. To lower the memory consumption and to enable parallel processing, *iMoon* partitions a point cloud into smaller sub-clouds, and execute feature matching operations only on selected partitions.

Density-based Model Partitioning. *iMoon* partitions a 3D point cloud into smaller ones following two rules. First, each partition includes 3D points that are created based on the image features extracted from a limited number of photos. Second, the area covered by each partition should not be too small. In practice, we define the minimum length and width of the area covered by each partition. An example is visualized in Figure 3.

Fingerprint-based Partition Selection. *iMoon* selects partitions for feature matching as described in Algorithm 1. It first applies K-Nearest Neighbour algorithm (k-NN) to acquire a set of coarse locations based on Wi-

⁵By default, VisualSFM set the minimum number of common features between two matched images to be 15.

Figure 4: An example of photo selection. (a) After a coarse location (black dot at the center) is acquired, 7 photos within the radius (i.e., the green dots within the black circle) are checked. The red points represent the 3D points covered by these photos. The field of view of an image is represented by two lines and an arc between them. (b) 3 out of the 7 checked photos (marked with black) are selected for feature matching based on camera position and facing direction.

Fi fingerprinting. After that, *iMoon* selects the partitions that cover these coarse locations, and then searches within the selected partitions for photos that are taken close to the coarse locations. In practice, the search space is defined by radius. To minimize the processing time while maintaining the performance, *iMoon* chooses a subset of photos within each radius as candidates for feature matching with the query photo. The candidates are chosen as described below. If the search result contains more than one photo taken from the same location and having the same facing direction, only one of these photos is selected. In case too few photos are returned, the radius will be increased to find more photos satisfying the requirements. Figure 4a and 4b depicts an example of photo selection based on the rules mentioned above.

Identifying Location and Facing Direction. After a set of candidate photos are returned, *iMoon* executes feature matching between each candidate photo and the query photo. The processing of photo pairs are conducted in parallel. If enough matches are found, the estimates of camera position and rotation will be returned. The facing direction of a camera is derived from the estimated camera rotation. It’s possible that several positions and rotations are returned, since several partitions were selected during the previous fingerprinting step and more than two of them contain enough common features for registering the query photo. In practice, the location of the user is estimated as the midpoint among all the returned camera positions, and the user’s facing direction is defined by averaging the camera angle offsets.

4.2 Compiling Navigation Meshes

After locating the end user, we need to calculate a navigation path for her, for which a navigation mesh is fundamental. Given a 3D point cloud of indoor environment, some points may represent obstacles, e.g., tables, chairs and walls. *iMoon* compiles a navigation mesh from a point cloud by extracting obstacles from these points, treating the rest of the blank areas as traversable areas. It is implemented through the following four steps.

- 1) *iMoon* filters out the points related to floor and ceiling by simply removing the points with the z-axis value, which represents the height, lower or higher than certain thresholds. In our experiment, we set the threshold to be 10 cm for the floor and 2 m for the ceiling. The rest of 3D points are projected directly onto the ground plane to acquire a 2D point set.

- 2) When there are user trajectories available, pedestrian paths will be extracted from these trajectories. A pedestrian path includes a sequence of points. Assuming that there are no obstacles like walls and tables along the path, all the points along the path can be removed from the 2D point set obtained in the previous step. As the path width is undefined, we assume that points included in a path always stay on the central line of the path. Thus *iMoon* also removes the neighbouring points that are close to the central line. In practice, we set the distance threshold to be 25 cm, assuming that the minimum path width is 50 cm.

- 3) *iMoon* applies the algorithm proposed by Duckham [9] to generate non-convex polygons out of the 2D points derived in the previous step. The basic idea of the algorithm is to wrap a set of points with a polygon, specifying which points belong to the outline and which lie inside the polygon. One polygon represents one obstacle. Accordingly, all the 2D points must be divided into groups, with each group corresponding to one obstacle. In practice, *iMoon* groups the points if the distance between any two points is less than a threshold, e.g., 50 cm.

- 4) After extracting the shape contours, *iMoon* extrudes all 2D shapes towards the z-axis to form solid meshes. The generated navigation meshes are saved in a Wavefront Object (.obj) file.

4.3 Generating Navigation Instructions

Based on the navigation meshes described in Section 4.2, *iMoon* implements an A* like path finding algorithm using RecastDetour library⁶. For each navigation path, *iMoon* generates turn-by-turn navigation instructions for users. As shown in Figure 5, an instruction comprises three parts: 1) a photo that shows the place where the given instruction is valid, 2) a piece of text that tells the user to go straight or make a turn and approximately how many meters to go until the next waypoint, and 3) an arrow sign that indicates which path to take next.

The photo is selected from the photo set used as input for 3D modelling. The selected photo is expected to cover the waypoint and to be clear enough to understand the surroundings. In our prototype, the photo is selected based on the following four steps.

- 1) Search for the 3D point that is closest to the 3D location of the waypoint in question, and find all the photos that contain features corresponding to this point. If less than N photos (N is set to 5 by default) are found, skip 2).

- 2) Sort the photos selected in 1) by the distance between the camera position and the waypoint, and for the next step select the first N photos with the shortest distances.

- 3) Sort the selected photos by the relative rotation regarding the navigation route. Its purpose is to find the photos that are taken facing the closest angle as the user will traverse the path. Thus, the best angle is facing the same way

⁶<https://github.com/memononen/recastnavigation>

Figure 5: A screenshot of the *iMoon* Android app in navigation mode. The upper part includes a mini map with the color information of each 3D point reserved. The navigation route (blue line) drawn on the map includes 3 waypoints between the source and the destination. The lower part gives a preview of the navigation instructions when the user arrives at the waypoint marked by a red dotted circle. The current position and facing direction of the user (in green) are updated on the move. In addition to the textual instruction, the selected image shows the expected view, and the arrow indicates the path to take.

as the user walks. After sorting, we take the first M photos (M is set to 3 by default) for the next step.

4) Sort the photos selected in 3) by the distance between the 3D point (defined in the first step) projection on the photo and the center of the photo. The photo with the closest center to the 3D point (thus also to the waypoint) is selected to form the navigation instruction that is given at the waypoint.

Once *iMoon* obtains a suitable photo, it draws directional arrows on it and renders textual instructions. Assume that the current location of a user is L_C , and the direction of the user is denoted by a vector $\vec{a}$. There will be 2 arrows drawn on the image: between L_C and an upcoming waypoint L_W (the one that was used to find the navigation photo), and between L_W and the next waypoint L_{W2} . Since the model building step extracted rotation matrices for each input image, *iMoon* calculate 2D projections of L_C , L_W and L_{W2} , and draw arrow shapes on the selected image by connecting these points. We denote the vector from L_C to L_W by $\vec{b}$, and the one from L_W to L_{W2} by $\vec{c}$. *iMoon* generates two textual instructions based on the angle between $\vec{a}$ and $\vec{b}$, and that between $\vec{b}$ and $\vec{c}$, respectively. According to the value of the angle, denoted by $\angle A$, *iMoon* provides textual instructions as defined in Eq. (1).

$$\text{Text} = \begin{cases} \text{'Go straight'}, & -T \leq \angle A \leq T \\ \text{'Turn left'}, & 180^\circ + T < \angle A < 360^\circ - T \\ \text{'Turn right'}, & T < \angle A < 180^\circ - T \\ \text{'Go back'}, & 180^\circ - T \leq \angle A \leq 180^\circ + T \end{cases} \quad (1)$$

where T is a threshold in degrees (set to 15° by default). The instruction also contains the information about the distance between L_C and L_W .

Data Set	Content	Usage	Dates
A	2,197 photos, 119 user trajectories, 4,239 Wi-Fi fingerprints	Building initial 3D models	Aug. 2014
B	355 photos, 31 user trajectories, 1,181 Wi-Fi fingerprints	Updating 3D models	Sep. 2014 - Mar. 2015
C	2,552 photos, 5,420 Wi-Fi fingerprints (Set A $\cup$ B)	Analyzing the quality of the generated 3D models	Aug. 2014 - Mar. 2015
D	2,220 groups of photos and Wi-Fi fingerprints	Performance evaluation of indoor localization	Sept.-Oct. 2014

Table 1: Summary of the data sets collected during our field study.

5. EVALUATION

We implemented a prototype of *iMoon* to evaluate the feasibility of building an indoor navigation system using crowd-sourced photos and sensor data. Since August, 2014, we have been carrying out a field study on the ground floor of the computer science building in our campus (referred to as **CS hall** in the rest of the paper). According to the field study, the functionalities of *iMoon* have proved to work properly. Therefore, we will focus on the performance evaluation of *iMoon*, and will introduce in this section the methodology, the experimental data sets, and the experimental results, including a short summary of feasibility study.

5.1 Methodology

iMoon can be used for locating users, calculating navigation routes, and providing turn-by-turn navigation instructions. Performance of *iMoon* depends on the performance of the localization, the accuracy of the navigation meshes used for path planning, and the accuracy of the navigation instructions. Because the navigation meshes are generated from the sensor-enriched 3D models (cf. Section 3), the accuracy of the navigation meshes is determined by the quality of the 3D models in use. Similarly, the accuracy of the navigation instructions depends on the accuracy of the estimates of user's position and facing direction, which also relies on the quality of the 3D models. Therefore, for performance evaluation of *iMoon*, we focus on the quality of the 3D models and evaluate it using the following metrics: the accuracy of the indoor layout extracted from the 3D models, and the performance of the 3D-model-based indoor localization.

We aim at answering the following questions based on performance measurement: What kind of crowdsourced data is desired for building high-quality 3D models of indoor environments for the purpose of indoor navigation? Does *iMoon* outperform the state-of-the-art indoor localization techniques? What is the impact of the input for 3D modelling on the localization performance?

5.2 Data Collection

We collected data in the CS hall, which is approximately $1,100 m^2$, covering a library, a cafeteria, and a corridor connecting over ten lecture rooms and offices. During weekdays,

Figure 6: 185 measurement points where data set D was collected. They were distributed in the cafe (black circle), corridor (red triangle) and library (blue square).

more than 500 students and staffs visit the CS hall on daily basis. The indoor scenes of the cafeteria change frequently due to promotions and decorations. During the field study, the construction work of a new food store started in March, 2015. Our data sets, as summarized in Table 1, include the data collected both before and after these changes occurred.

To bootstrap the system, we recruited 5 volunteers in August, 2014 and asked them to collect photos in the CS hall. The volunteers can choose by themselves where to take photos. For each location they chose, they were requested to take 5 to 10 photos with the camera facing different directions. Meanwhile, the *iMoon* mobile app was configured to record Wi-Fi fingerprints and the readings of accelerometer and gyroscope while the volunteers were walking in the CS hall. The data collection started immediately after the first photo was taken, and can be stopped manually. Over a period of 3 weeks, the volunteers collected in total 2,197 photos and 4,239 groups of Wi-Fi fingerprints along 119 walking traces (referred to as **data set A** in Table 1).

After the system was initialized with the data set A, the volunteers were invited to test the indoor navigation services. During the trial period, another 355 photos and 1,181 Wi-Fi fingerprints along 31 walking traces were collected (referred to as **data set B**) till March 2015. Accordingly, the 3D models available at the end of March 2015 were built from the union of the data set A and B (referred to as **data set C**).

During September and October 2014, we collected **data set D** for the performance evaluation of indoor localization⁷. The data set D includes 2,220 photos taken from 185 measurement points in the CS hall. As shown in Figure 6, the measurement points are distributed evenly across the walkable area in the hall.

We used a self-built toolkit, as shown in Figure 7a, to obtain the ground truths corresponding to the data set D. At each measurement point, we took photos and collected Wi-Fi fingerprints using the Android phone placed on the tripod. After taking one photo, the camera was rotated horizontally by 30 degrees before taking another one. More specifically, before taking the first photo at each measurement point, the camera was facing the same direction defined as the baseline direction. After that, the camera was rotated by 30 degrees to take the next photo. We repeated the process until the camera was facing the baseline direction again. At each measurement point, we collected 12 photos. Additionally, we used a laser rangefinder to measure the distance to each reference point. Given the measurements from the distance and locations of the reference points, we calculated the

⁷Note that the 3D models built from the data set A were used for performance analysis in this case.

Figure 7: Evaluation toolkit. The left-hand figure shows a self-built toolkit that consists of a laser rangefinder, a protractor, an Android phone running *iMoon* mobile app, and a tripod as a stand. The right-hand figure demonstrates how trilateration is applied to determine the location of the toolkit based on measurements of the distance to each reference point.

actual location of the camera based on trilateration, as illustrated in Figure 7b. For each photo, the actual location and the facing direction of the camera was recorded and used as the ground truth for accuracy analysis.

5.3 Accuracy of Navigation Mesh

We built an initial 3D model of the CS hall using data set A in August 2014. Following the SfM pipeline described in Section 3.1, the initial 3D model (as shown in Fig. 8b), which includes 253,839 3D points, was successfully built from 1,968 (out of 2,197) photos in data set A. Based on the generated 3D models and the user trajectories included in data set A, *iMoon* compiled a navigation mesh as described in Section 4.2. The navigation mesh is visualized in Figure 9a. For comparison, a navigation mesh generated from the initial 3D model only is also shown in Figure 9a. We can see that the pedestrian paths, especially the one along the corridor, are better reconstructed when user trajectories are used for compiling navigation meshes.

From September 2014, the initial models have been updated constantly using the photos in data set B. As illustrated in Figure 8b and 8c, the 3D models have been successfully updated to reflect a significant change of indoor layout caused by the ongoing construction work of a Subway store. The actual indoor scenes before and after the construction work started are shown in Figure 8e and 8f, respectively.

Compared with the official floor plan released in 2008 (c.f. Figure 8a), the 3D models generated by *iMoon* well reconstruct the outline of the CS hall. To evaluate the accuracy of the reconstructed outline, we calculated the Euclidean distances of walls surrounding the CS hall based on the 3D models, and measured the ground truth using a laser rangefinder. After that, we calculated the distance error as defined in [23]. Specifically, the distance error for a correctly detected segment is defined as the average distance between the detected segment and its associated ground truth segment. The overall distance error, denoted by E_D , is calculated below.

$$E_D = \frac{\sum_{i=1}^N [\text{length}(s_i) E_d(s_i)]}{\sum_{i=1}^N \text{length}(s_i)}, \quad (2)$$

Figure 8: Comparison of the generated 3D models with the actual layout of the CS hall.

Figure 9: Navigation meshes generated from 3D models.

where N is a number of correctly detected segments, and $E_d(s_i)$ is the distance error for a correctly detected segment s_i . According to the results, *iMoon* is able to reconstruct the outline of the CS hall with an overall distance error of 0.502 meters.

Based on the 3D models built in March, 2015, *iMoon* compiles a navigation mesh including 215 obstacles. The navigation mesh is visualized in Figure 9b. We calculate the average location error of obstacles according to Eq. (3). Denote the coordinates of an obstacle in the mesh by (x_i, y_i) , and its actual coordinates by (X_i, Y_i) , the average location error of obstacles, denoted by $E_{obstacles}$, can be calculated as

$$E_{obstacles} = \sum_{i=1}^N \sqrt{(x_i - X_i)^2 + (y_i - Y_i)^2} / N \quad (3)$$

where N is the number of obstacles. For the navigation mesh generated from the 3D models available in March, 2015, its average location error is 0.78 meters, which indicates a reasonably high accuracy of the navigation mesh.

Our experiment results have shown that the data set A, which includes 2,197 photos collected from locations that are nearly evenly distributed in the area, is enough to build a 3D model of the CS hall with reasonably good quality. Concerning the complexity of indoor structure and the diversity of indoor scenes, the number of photos needed for building a good-quality 3D model varies with indoor environments. Furthermore, the quality of 3D models depends also on the manner of data collection, including where to take photos and what to be captured by the photos.

To analyze the impact of data collection, we sampled the photos in data set C in 5 different ways, and compared the results of the 3D models built from each subset accordingly. The 3D models created from all the photos in data set C was used as the baseline. First, we grouped the photos by location and facing direction, and then evenly selected 50% of the photos. The set of sampled photos is referred to as **Sample 1**. Similarly, we evenly sampled 25% of the photos from data set C to form **Sample 2**. Compared with the data set C, the photos in Sample 1 and Sample 2 were taken with lower density but still trying to cover the whole space. To emulate the data collection in a more random manner, we also built **Sample 3** and **Sample 4** by selecting 50% and 25% of photos from data set C randomly, regardless of location or facing direction. In principle, in order to use as few photos as possible for building 3D models while maintaining the performance of localization, the input photos are expected to have as little overlap as possible while having sufficient common features to be matched. Depending on various factors such as the settings of the cameras and the texture of the walls, it is not possible to tell exactly how many photos are needed. For comparison, we manually selected 20% of the photos from data set C by removing photos that captured similar scenes. The selected photos form **Sample 5**.

The number of 3D points generated from the 5 samples are listed in Figure 10. From the figure we can observe that as less photos are used for 3D modelling, the output 3D models are more prone to be fragmented, which is clearly illustrated in cases where Samples 2, 3, and 4 are used as

Figure 10: Comparison of the number of 3D points included in the 3D models built from different samples. The different color in each bar represents the number of points included in each fragment of the 3D model in question.

Area	No. Measurement Points	No. of Photos	Hit rate		
			Image	Wi-Fi	<i>iMoon</i>
Cafe	78	936	91%	100%	91%
Corridor	37	444	100%	100%	100%
Library	70	840	98.6%	100%	97.1%
Total	185	2,220	95.7%	100%	95.1%

Table 2: Comparison of hit rate between image-based localization, Wi-Fi fingerprinting and the localization algorithm implemented in *iMoon*.

input. On the other hand, the 3D models built from Sample 1 remain almost as intact as the ones from data set C, from fragmentation perspective. Compared with the largest 3D models generated from Samples 3 and 4, the biggest 3D model created from Sample 5 contains more 3D points. This is because the scenes captured by different photos in Sample 5 are expected to have less overlaps. The impact on the density of 3D points will be discussed in Section 5.4.3.

5.4 Performance of 3D-model-based Indoor Localization

We measured the performance of indoor localization with four metrics, namely, hit rate, location error, direction error, and response delay. **Hit rate** refers to the percentage of measurement points that can be located. When there is a hit, the accuracy of the result is measured with **location error**, and **direction error** if the result includes also user’s facing direction. The location error calculates the distance between the estimated location of the measurement point and the ground truth, while the direction error measures the angle offset from the actual facing direction. Additionally, **response delay** measures how much time it takes for the server to return the result. Note that the response delay of indoor localization depends on the computation complexity of the localization algorithm under test, as well as the computing power of the computing infrastructure in use.

iMoon combines Wi-Fi fingerprinting with image-based localization to achieve fast while accurate indoor localization. To provide an objective comparison with the state-of-the-art, we provided a reference implementation of the image-based localization (abbreviated as **Image**). The reference implementation of **Image** applies Breadth-first Search (BFS) to select candidates for feature matching, and tries to

Figure 11: Examples of non-matching inputs (a) (b) and a matching input (c) from one measurement point located in the corridor.

Figure 12: The numbers of measurement points where the indicated amount of matching inputs are identified at each point. Samples 1 and 5 are subsets of data set C in the library. Each color represents the number of matching inputs at a certain measurement point. For example, there are 15 (out of 37) measurement points in the corridor. At each of these points 6 matching inputs can be found.

search through the whole 3D point cloud instead of selected partitions to get the best possible result. Both *iMoon* and the reference implementation of **Image** were tested with the 3D models built from the photos in data set A. In addition, to provide a comparison with the Wi-Fi fingerprinting based approaches, we utilized the Wi-Fi fingerprints included in data set A to implement a standalone Wi-Fi fingerprinting solution (abbreviated as **Wi-Fi**). We then used data set D to test the performance of both *iMoon* and **Wi-Fi**. The experimental results are presented in the following subsections.

5.4.1 Hit Rate

If a valid location can be returned by the localization algorithm, we consider the input as a **matching input** for the algorithm; otherwise, it is considered as a non-matching input. In data set D, 12 photos and the associated Wi-Fi fingerprints were collected from each measurement point. If at least one of the 12 groups of data was considered as matching input, we count the corresponding measurement point as a locatable point. The hit rate is calculated as the number of locatable points divided by the total number of measurements points.

In Table 2 we compare the hit rate among *iMoon*, **Wi-Fi**, and **Image**. In general, the hit rate is reasonably high in all the three cases. Compared with **Wi-Fi**, the two image-based schemes (i.e., *iMoon* and **Image**) fail to locate around 5% of the measurement points. This is because of the changes of indoor scenes occurred after the initial 3D model was built. Compared with the corridor, the layout of the cafeteria and the library changed more frequently, which explains the relatively low hit rates in such areas. Furthermore, **Image** also achieves slightly higher hit rate than *iMoon*. This is because *iMoon* only selects a subset of model partitions for feature matching in order to lower the

Figure 13: Comparison of (a) location error and (b) direction error between *iMoon*, **Image**, and **Wi-Fi**. Note that **Wi-Fi** does not return user’s facing direction.

Figure 14: Comparison of location error between scenarios where the 3D models were built from data set C, Sample 1, and Sample 5, respectively.

response delay. The partitions filtered out by Wi-Fi fingerprints in *iMoon* may contain feature matches with the query photo.

Photos taken by different users at the same place may cover different scenes, due to the user’s facing direction and the configurations of the camera such as focal length. If the photo includes mostly the texture-less scenes, e.g., white walls, it is very likely that not enough distinguishable features can be used for feature matching. Thus, most likely, these photos will become non-matching inputs. Because our measurement points are distributed evenly in the CS hall and 12 photos facing different direction were taken at each point, it is inevitable that a certain number of photos were taken with the camera facing texture-less scenes. Two examples are shown in Figure 11a and 11b. Those photos provide much less features compared with the example of matching input shown in Figure 11c. This also explains why the experiment in the library has higher percentage of measurement points with more matching inputs than in the corridor, as shown in Figure 12. In practice, if the first photo taken by a user is a non-matching input, *iMoon* will request the user to face a different direction and then take a new photo for another try. When people try to locate themselves, they will very possibly take photos of distinguishable scenes instead of plain walls. Thus, the number of matching inputs can be expected to be higher in real cases.

5.4.2 Accuracy

As explained previously, we collected ground truth of locations and facing directions using own developed toolkit, and evaluated the accuracy of indoor localization based on the ground truth and the estimated location and facing direction (if available). As shown in Figure 13a and 13b, the accuracy of *iMoon* is very close to that of **Image**. In both cases, for at least 90% of the valid inputs, the location error is less than 2 meters, while the direction error is less than 6 degrees. Similar to the hit rate (cf. Table 2), the accuracy of **Image** is slightly different from that of *iMoon*, due to the pre-filtering of model partitions used in *iMoon*.

Figure 15: Location and facing direction errors in the cases where a whiteboard was brought into the scene and then moved towards the camera.

The result of **Wi-Fi** does not include an estimate of the facing direction. The location error of **Wi-Fi** is less than 5 meters in approximately 70% of the test cases and less than 10 meters in 90% of cases. Obviously, the accuracy of **Wi-Fi** is relatively low, compared with the image-based approaches. On the other hand, because the hit rate of **Wi-Fi** is higher, it is reasonable to fall back to the coarse locations provided by Wi-Fi fingerprinting, when the system fails to obtain a more accurate location based on photos.

5.4.3 Robustness

To investigate the impact of the input for 3D modelling on the accuracy of localization, we chose the 3D models built from **data set C** as a reference, and built another two models from **Samples 1** and **5**, respectively, for comparison. We chose the library as the testing environment, and selected the related photos from **data set D** for test.

As illustrated in Figure 14, the system achieves higher accuracy with the reference model than the ones built from Samples 1 or 5⁸. Recall that more test data become matching inputs with a denser point cloud, as shown in Figure 12. Hence, we can conclude that the positioning accuracy increases with the density of the point cloud. Nevertheless, the density does not increase if less photos are used for building the point cloud.

Objects like whiteboards and chairs may appear (disappear) or move around frequently in the building after the 3D models are built. Additionally, people may accidentally show up in front of the camera when photos are taken for localization. To evaluate how the localization accuracy is affected, we conducted a set of tests where we brought a whiteboard to the library and moved it towards our camera slowly, trying to block the whole camera view gradually. We took 18 photos from the same location while the whiteboard was moving as described in Figure 15a – 15c. After that, we used *iMoon* to locate the camera with each of these photos. With 17 out of the 18 photos, *iMoon* successfully identified the location with comparable accuracy, as shown in Figure 15d and 15e. *iMoon* failed to locate the camera only when the photo captured nothing more than the whiteboard. The localization accuracy remained at the same level as far as the location was successfully identified.

⁸The indoor navigation system performs better than average in the library, because the scenes in the library are more distinguishable than the corridor for example. That is why the results in Figure 14 are better than the ones shown in Figure 13a.

Figure 16: Comparison of response delay between indoor localization algorithms using the data set D as test data set.

5.4.4 Response Delay

We deployed *iMoon* on a server equipped with an Intel Xeon processor E5-2650 (8-core, 2.6GHz), 64GB RAM, and a Tesla K20C GPU. We measured the response delay using data set D as input, and compared the results between *iMoon*, **Wi-Fi**, and **Image**. Note that the testing data set includes both matching and non-matching inputs from the perspective of localization.

Figure 16a shows that the response delay of **Wi-Fi** is less than 0.6s for 87% of the input. For **Image**, as presented in Figure 16b, it takes more than 40s to generate a response (including the time spent in loading the 3D models). The loading time increases with the size of 3D models. According to our experiment, the loading time can be ignored for a 3D model built from less than 300 photos. However, it takes as long as 35s to load a 3D model built from 2,000 photos. Excluding the loading time, 41% of the tests take around 15s to complete the process of feature matching and camera pose estimation, while the rest of the tests take longer time due to selecting more images for matching.

iMoon supports 3D model partitioning and fingerprint-based partition selection. As shown in Figure 16c, the response delay, including loading time, is decreased to 4s for 73% of the tests when each model partition contains features extracted from less than 100 photos. The response delay excluding the loading time is on average 3.85s with standard deviation of 1.5s. Note that the response delay can be further decreased by utilizing more powerful machines.

5.5 Summary

From the experiments and analysis presented above, we can reach the conclusion that it is feasible to build a 3D-model-based indoor navigation system with reasonably high performance, using only the photos and sensor data collected from smartphones. The quality of 3D models, nevertheless, is affected by multiple factors, including the number of photos used for generating the 3D models, and the coverage and the facing direction of the input photos. Although the coverage of crowdsourced photos may be limited to certain areas of interest, with more and more photos becoming available, there is an increasing chance of producing high-quality 3D models for the purpose of indoor navigation. Furthermore, the accuracy and response delay of *iMoon* are decent and acceptable compared to the state-of-the-art.

6. DISCUSSION AND FUTURE WORK

Varying quality of crowdsourced data. Although crowdsourcing provides a relatively cheap way of collecting data, there is always an uncertainty in the quality of the available data. In other words, it is hard to know in advance how much data contributed by the crowd is actually usable. For example, in places of interest, people are usually much more interested in taking photos of landmarks

rather than ordinary scenes like white walls. This can be observed from photos available on Internet photo sharing websites like Flickr and Instagram. As a result, the crowdsourced photos may not cover every corner of the space of interest, which means the 3D models generated from the crowdsourced photos may have “holes” in them and some models may be disconnected from each other, causing the fragmentation issue. Although *iMoon* can still provide indoor navigation with slightly compromised accuracy using “non-perfect” 3D models, it is worth investigating incentive schemes that would encourage and guide users to provide data for fixing the “holes”. Developing such an incentive scheme is part of our future work. We will also investigate the feasibility of utilizing crowdsourced user trajectories for connecting 3D models in order to solve the fragmentation issue. We believe that with these new features, *iMoon* will be able to provide increasingly better service.

Tracking user trajectories using non-perfect maps.

In navigation mode, once the initial location is identified, *iMoon* starts tracing users using the motion sensors on smartphones, whereas for the purpose of trace calibration users are still requested to take photos while making turns. It is possible to track users based on the sensor data only. An example solution is particle filtering based dead reckoning [15, 25]. The challenge to such solution comes from the fact that the navigation meshes built from crowdsourced data may not be complete or 100% accurate. This issue should be taken into account while applying the existing algorithms for indoor navigation based on crowdsourced maps.

Combination of latency-constrained and compute-intensive operations. Generally, building 3D models requires heavy computation. Compared with 3D modelling, calculating locations or generating navigation instructions demands less computing power, whereas it is much more sensitive to delay. To meet the requirements of low latency and computing power, we propose to deploy *iMoon* in a hybrid cloud infrastructure. For example, the initial 3D models are first generated on public clouds, like Amazon EC2, which have more computing power, and then transferred to a private cloud that is much closer to the end users. After that, the real-time navigation services will be provisioned from the local cloud, e.g., a cloudlet [28] or a Nokia RACS server.

Measurement of user experience. In the future, we will deploy *iMoon* using hybrid cloud infrastructure in more dynamic environments such as shopping malls, and evaluate the performance of *iMoon* in realistic scenarios. We plan to conduct a survey between test users and ask them to score the system from the perspective of user experience, including the accuracy, response delay and user-friendliness of the services.

Utilizing text and sign information from crowdsourced images. We plan to extend our system to support descriptive search, e.g., by place name or room num-

ber. Photos taken from indoor environment usually contain signs and descriptive texts, it makes sense to utilize such information for place search to improve usability. We plan to apply sign recognition and optical character recognition techniques to extract information from the input photos. If a sign or word is detected from a photo, we can first acquire pixel coordinates of the sign/word in the photo. Then, a set of features can be extracted from the photo. By comparing the coordinates of each feature point with the photos of the recognized sign/word, the closest feature point to the sign/word can be found and set as the location of the sign/word in question.

7. RELATED WORK

3D-Model-based Indoor Mapping. 3D modelling has been widely used for indoor mapping. Conventionally, 3D models for indoor mapping are generated from the data captured by laser scanners and/or depth cameras through war-driving. For example, the system presented in [23] obtained 3D point cloud of indoor environment from laser scanners positioned in multiple locations, and extracted 2D floor plan from the 3D point cloud. KinectFusion [21] implemented real-time 3D surface reconstruction of room-sized scenes based on the depth data streamed from a moving Kinect sensor. The recently developed SLAM (Simultaneous Localization and Mapping) based indoor mapping tools, such as Google Cartographer and Xsens Scannet [6], are equipped with inertial measurement units in addition to laser scanners and depth cameras. The Cartographer-like backpack systems can automatically create floor plans when the backpacker-wearers walk inside the buildings.

Differently from SLAM, SfM techniques enable 3D modelling of indoor and outdoor environments using unordered 2D photos. These photos can be taken by ordinary devices like smartphone cameras. Based on SfM, Agarwal et al. [3] constructed 3D models of Rome city from 150K photos found from Internet photo sharing sites. Furukawa et al. [11] used SfM and multi-view stereo for reconstructing and visualizing an entire house interior. Previous work has also tried to solve the technical issues, such as the disconnection of the photo-based 3D point clouds [20] and the connection of wall segments based on motion sensor data [12]. *iMoon* also adopts SfM for 3D modeling of indoor environments. Depending on the input, the generated 3D models may not contain all the pedestrian path information. *iMoon* solves this issue by complementing the 3D models with crowdsourced user trajectories.

Indoor Localization and Navigation. Previous work has proposed to utilize fingerprints of Wi-Fi [4], magnetic field [7], Bluetooth [5] for finding locations in indoor environments. Fingerprinting-based approaches require a training data set comprising measurements at known locations in the space of interest. Since labelling fingerprints is labour-intensive, Zee [25] proposed automatic inferencing of location based on the combination of inertial sensor information and the constraints imposed by the map. The inertial sensor information was collected while simultaneously performing Wi-Fi scans. Similarly, Nguyen et al. [22] applied dead reckoning approach to track users in indoor environment. Due to the error accumulation of the dead reckoning approach, Nguyen et al. utilized active learning for automatically identifying strategically important locations which should be labelled manually. Rather than manually labelling Wi-Fi fin-

gerprints, *iMoon* utilizes random photos taken along user trajectories for automatic calibration, utilizing the feature of image-based localization.

Image-based localization system with the use of 3D SfM model allows users to locate themselves by simply taking photos from where they are. Photos are localized with high accuracy based on image feature matching and camera pose estimation. However, this process is slow due to heavy computation. To accelerate this process, Lu et al. [19] proposed to use the visual words and the approximate nearest neighbour methods. Similarly, Sattler et al. [27] proposed a framework for efficient 2D-to-3D matching based on visual vocabulary quantization and a prioritized correspondence search. Differently from the previous work, *iMoon* utilizes Wi-Fi fingerprints instead of image features for reducing the search space, taking the advantage of fast response of Wi-Fi fingerprinting and high accuracy of image-based localization.

In addition to location, a walking direction of user is also useful for creating and updating navigation instruction. Google Project Tango development kit⁹ combines 3D motion tracking with depth sensing to allow mobile devices to know where it is and how it moves through the space. *Walk-Compass* [26] detected the walking direction within a few steps using sensors available on smartphones, while Husen et al. [14] employed a dense Wi-Fi infrastructure to determine the orientation. In our work, user's facing direction is considered to be the same as the facing direction of the smartphone camera, and is obtained from the camera pose provided by image-based localization.

Regarding navigation instructions, a conventional 2D map based navigation system provides instructions including abstract paths [29] and texts [16]. With image-based 3D models, *iMoon* is able to generate navigation instructions that include visual hints. In practice, *iMoon* shows what the user is expecting to see in front by selecting photos from the input of 3D models based on user's position and facing direction.

8. CONCLUSION

We presented *iMoon*, a low-cost and whole-system solution for indoor navigation in this paper. *iMoon* is partially built on top of several existing techniques, e.g., SfM and fingerprinting. Nonetheless, *iMoon* brings new functions and addresses several technical challenges to enable mobile crowdsensing-based indoor navigation. It utilizes crowdsourced photos to build the 3D models of the indoor space of interest, and detects pedestrian paths from crowdsourced user trajectories. It also enables fast localization by taking advantage of the high hit-rate and low response delay of Wi-Fi fingerprinting. The new elements introduced into *iMoon* make it functional and perform well, as demonstrated in the field study. *iMoon* would be a good primer for any system considering providing indoor navigation services based on crowdsourced photos and sensor data.

9. ACKNOWLEDGMENTS

We thank the anonymous reviewers and Prof. Xiaofan Jiang for their insightful comments and suggestions. We thank our group members and volunteers, Emil Kreutzman, Iivari Äikäs, Teemu Kämäräinen, Morteza Neishaboori and Yinan Zong. This work was supported by the Academy of Finland, grant number 278207 and 268096.

⁹<https://www.google.com/atap/project-tango/>

10. REFERENCES

- [1] Siftgpu: A gpu implementation of scale invariant feature transform (sift). <http://cs.unc.edu/~ccwu/siftgpu/>.
- [2] Visualsfm : A visual structure from motion system. <http://ccwu.me/vsfm/>.
- [3] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *ACM Commun.*, 54(10):105–112, 2011.
- [4] P. Bahl and V. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *Proc. of INFOCOM 2000*, pages 775–784.
- [5] L. Chen, L. Pei, H. Kuusniemi, Y. Chen, T. Kroger, and R. Chen. Bayesian fusion for indoor positioning using bluetooth fingerprints. *Wireless Personal Commun.*, 70(4):1735–1745, 2013.
- [6] J. C. Chow. *Multi-Sensor Integration for Indoor 3D Reconstruction*. PhD thesis, University of Calgary, 2014.
- [7] J. Chung, M. Donahoe, C. Schmandt, I.-J. Kim, P. Razavai, and M. Wiseman. Indoor location sensing using geo-magnetism. In *Proc. of MobiSys’11*, pages 141–154, New York, NY, USA.
- [8] J. Dong, Y. Xiao, Z. Ou, and A. Ylä-Jääski. Utilizing internet photos for indoor mapping and localization – opportunities and challenges. In *Proc. of SmartCity’15*, pages 1–6, Hong Kong, China.
- [9] M. Duckham, L. Kulik, M. Worboys, and A. Galton. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recogn.*, 41(10):3224–3236, 2008.
- [10] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day. In *Proc. of ECCV’10*, pages 368–381, Berlin, Heidelberg.
- [11] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Reconstructing building interiors from images. In *Proc. of ICCV’09*, Kyoto, Japan.
- [12] R. Gao, M. Zhao, T. Ye, F. Ye, Y. Wang, K. Bian, T. Wang, and X. Li. Jigsaw: Indoor floor plan reconstruction via mobile crowdsensing. In *Proc. of MobiCom’14*, pages 249–260, New York, NY, USA.
- [13] Y. Gu, A. Lo, and I. Niemegeers. A survey of indoor positioning systems for wireless personal networks. *IEEE Commun. Surveys Tuts*, 11:13–32, 2009.
- [14] M. N. Husen and S. Lee. Indoor human localization with orientation using wifi fingerprinting. In *Proc. of ICUIMC’14*, pages 109:1–109:6, New York, NY, USA.
- [15] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao. A reliable and accurate indoor localization method using phone inertial sensors. In *Proc. of UbiComp’12*, pages 421–430, New York, NY, USA.
- [16] J. Link, P. Smith, N. Viol, and K. Wehrle. Footpath: Accurate map-based indoor navigation using smartphones. In *Proc. of IPIN’11*, pages 1–8, Guimaraes, Portugal.
- [17] H. Liu, H. Darabi, P. Banerjee, and J. Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Trans. Syst., Man, Cybern.*, 37(6):1067–1080, 2007.
- [18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.
- [19] G. Lu and C. Kambhamettu. Image-based indoor localization system based on 3d sfm model. In *Proc. of SPIE, Intelligent Robots and Computer Vision*, 2014.
- [20] R. Martin-Brualla, Y. He, B. C. Russell, and S. M. Seitz. The 3d jigsaw puzzle: Mapping large indoor spaces. In *Proc. of ECCV’14*, pages 1–16. Zurich, Switzerland.
- [21] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. of ISMAR’11*, pages 127–136, Basel, Switzerland.
- [22] L. T. Nguyen and J. Zhang. Wi-fi fingerprinting through active learning using smartphones. In *Proc. of UbiComp’13 Adjunct*, pages 969–976, New York, NY, USA.
- [23] B. E. Okorn, X. Xiong, B. Akinci, and D. Huber. Toward automated modeling of floor plans. In *Proc. of 3DPVT’10*, Paris, France.
- [24] Z. Ou, J. Dong, S. Dong, J. Wu, A. Yla-Jaaski, P. Hui, R. Wang, and A. Min. Utilize signal traces from others? a crowdsourcing perspective of energy saving in cellular data communication. *IEEE Trans. Mobile Comput.*, 14(1):194–207, 2015.
- [25] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: Zero-effort crowdsourcing for indoor localization. In *Proc. of Mobicom’12*, pages 293–304, New York, NY, USA.
- [26] N. Roy, H. Wang, and R. Roy Choudhury. I am a smartphone and i can tell my user’s walking direction. In *Proc. of MobiSys’14*, pages 329–342, New York, NY, USA.
- [27] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *Proc. of ICCV’11*, pages 667–674, Barcelona, Spain.
- [28] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Comput.*, 8(4):14–23, 2009.
- [29] A. Serra, D. Carboni, and V. Marotto. Indoor pedestrian navigation system using a modern smartphone. In *Proc. of MobileHCT’10*, pages 397–398, New York, NY, USA.
- [30] N. Snavely, I. Simon, M. Goesele, R. Szeliski, and S. Seitz. Scene reconstruction and visualization from community photo collections. *Proc. of the IEEE*, 98(8):1370–1390, 2010.
- [31] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: Unsupervised indoor localization. In *Proc. of MobiSys’12*, pages 197–210, New York, NY, USA.
- [32] C. Wu. Towards linear-time incremental structure from motion. In *Proc. of 3DV’13*, pages 127–134, Seattle, USA.
- [33] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *Proc. of CVPR’11*, pages 3057–3064, Washington, DC, USA.