Indoor Localization via Multi-Modal Sensing on Smartphones

Han Xu
CSE, HKUST
hxuaf@cse.ust.hk

Zheng Yang
TNLIST, Tsinghua
yang@greenorbs.com

Zimu Zhou
TIK, ETH Zurich
zimu.zhou@tik.ee.ethz.ch

Longfei Shangguan
CS, Princeton
longfeis@cs.princeton.edu

Ke Yi
CSE, HKUST
yike@cse.ust.hk

Yunhao Liu
TNLIST, Tsinghua
yunhao@greenorbs.com

ABSTRACT

Indoor localization is of great importance to a wide range of applications in shopping malls, office buildings and public places. The maturity of computer vision (CV) techniques and the ubiquity of smartphone cameras hold promise for offering sub-meter accuracy localization services. However, pure CV-based solutions usually involve hundreds of photos and pre-calibration to construct image database, a labor-intensive overhead for practical deployment. We present *ClickLoc*, an accurate, easy-to-deploy, sensor-enriched, image-based indoor localization system. With core techniques rooted in semantic information extraction and optimization-based sensor data fusion, *ClickLoc* is able to bootstrap with few images. Leveraging sensor-enriched photos, *ClickLoc* also enables user localization with a single photo of the surrounding place of interest (POI) with high accuracy and short delay. Incorporating multi-modal localization with Manifold Alignment and Trapezoid Representation, *ClickLoc* not only localizes efficiently, but also provides image-assisted navigation. Extensive experiments in various environments show that the 80-percentile error is within $0.26m$ for POIs on the floor plan, which sheds light on sub-meter level indoor localization.

Author Keywords

Indoor Localization; Smart Phone; Multi-Modal Data

ACM Classification Keywords

H.3.4. Information Storage and Retrieval: Systems and Software

INTRODUCTION

Accurate indoor localization on unmodified smartphones is a key enabler for near-future commercial location-based services such as customer navigation in supermarkets, targeted advertisements in shopping malls, and augmented reality in public places. Towards this envision, we need localization technologies that achieve high accuracy (*e.g.* sub-meter) and are easily deployable at scale.

Crowdsourced WiFi-based fingerprinting [11, 35] and inertial-based pedestrian dead-reckoning [15, 22] are two mainstream easy-to-use approaches, which usually yield meter-level accuracy. While meter-level accuracy is sufficient to localize or navigate a customer within a shopping mall, sub-meter level accuracy is helpful to determine which aisle a customer is facing within a particular grocery store, to guide a user to the shelf to fetch a desired book in the library, and to provide detailed information when a customer stands in front of a painting in an art gallery [25]. As a promising alternative, image-based localization, primarily adopted in robotics, usually offers sub-meter or even centimeter level accuracy [19], at the cost of sophisticated image database construction [20] or dedicated infrastructure [31].

We observe an opportunity for an easy-to-deploy image-based smartphone indoor localization system, that preserves the sub-meter accuracy, yet dramatically reduce the overhead involved in pure CV-based image database construction. The key idea is to leverage *minimal sensor-enriched photos* to map the image-generated POI model into the physical space, a major overhead for image database construction, which conventionally involves hundreds of photos for a single POI [6]. Moreover, the POI-based (as opposed to interior structure-based [1]) image database construction facilitates incremental database construction and updating, an indispensable feature for crowdsourcing the labor of database construction to unprofessional users. The sensor enriched photos also hold potential to speeds up the localization process by exploiting the inherent geometric constraints derived from sensor measurements.

Based on the above intuition, we propose *ClickLoc* (indoor localization with a single click), an easily deployable and highly accurate image-based indoor localization system with sensor fusion. In designing *ClickLoc* into a practical localization system, two main challenges need to be addressed: (1) *How to robustly map the relative point cloud generated by images into physical coordinates during the training stage?* (2) *How to maximally ease user localization during the operating stage?* To robustly construct the image database in real

UbiComp'16, September 12–16, 2016, Heidelberg, Germany.
© 2016 ACM. ISBN 978-1-4503-4461-6/16/09$15.00
DOI: http://dx.doi.org/10.1145/2971648.2971668

indoor environments with minimal number of photos, we utilize the semantic information via Indoor Geometry Reasoning [14], and design a set of optimization techniques to derive accurate transformation (scaling, rotation, and translation) from multi-sensor data. The lightweight mapping mechanism requires *as few as two photos* to add one POI into the database (10-20 are suggested in practice). To speed up the localization procedure, we first estimate a rough location with sensor data (WiFi and inertial measurements) using manifold alignment [8] and trapezoid overlapping, which dramatically reduce the search space for image matching afterwards. The images of the candidate POIs are then matched with the query image to return the estimated location, which requires *only a single photo* taken from the user.

We fully prototype *ClickLoc* on Android platforms and conduct extensive experiments in two large complex indoor environments including a shopping mall and a railway station. Evaluations demonstrate that *ClickLoc* achieves a 50-percentile error of $0.17m$ and a 80-percentile error of $0.26m$ for POIs on the floorplan, and a 50-percentile error of $0.7m$ and a 80-percentile error of $1.0m$ for arbitrary POIs, which preserves the accuracy of image-based localization schemes, yet incurs far less overhead during image database construction.

The key contributions are summarized as follows.

- **Bootstrap Database Construction with Few Images:** By exploiting both semantic information (from floorplan and photos) and geometric constraints (from inertial sensors), we estimate the transformation from the image space to the physical space without pre-calibrated images. Hence we significantly mitigate the cost of database construction, a primary overhead for image-based localization system.

- **Fasten Image-based Localization Process:** By implementing multi-modal localization incorporating Manifold Alignment (WiFi and floorplan) and Trapezoid Representation, we fasten the localization running time by $\times 1$ (fewer than $9s$ per query in our evaluation).

- **Sub-meter Level Accuracy with One Click:** Evaluations in two large complex indoor environments show that *ClickLoc* achieves sub-meter localization accuracy (a 50-percentile error of $0.7m$ for arbitrary tested POIs), which only requires the user to take and upload one photo to operate, which sets a new level for accurate, easy-to-use, and user-friendly indoor localization system.

In the rest of the paper, we first review the related work and preliminaries, followed by an overview and detailed design of *ClickLoc*. We then describe the evaluation of *ClickLoc* and discuss on the limitations before concluding this work.

RELATED WORK

The rich embedded sensors in smartphones have attracted extensive indoor localization research with one or multiple sensing modalities. Popular approaches including wireless [16], inertial sensors [9] and cameras [19]. As a sensor-enriched image-based smartphone indoor localization system, *ClickLoc* is closely related to the following threads of research.

Image-based Indoor Localization. Compared with its counterpart (*e.g.* wireless and inertial-based), image-based localization can easily yield sub-meter accuracy [19]. The key advantage is that the geometric relationships of indoor objects extracted from images using computer vision techniques are usually orders more accurate than those inferred by wireless or inertial based methods. The high accuracy makes image-based approaches a fit for robot localization and navigation [2]. Adopting pure image-based localization to smartphones incurs two limitations. (1) Hundreds of photos are required to extract geometric relationships of objects using *e.g.* Structure from Motion [28]. (2) The derived geometric relationships are *relative*. Therefore an accurate mapping between the image space and the physical space is indispensable. This is no problem with robots, which have precise control of their locations, yet a challenge for unprofessional users, who are often unaware of the location of their phone cameras. *ClickLoc* aims to overcome these limitations, and enables easy-to-use image-based localization on unmodified smartphones.

Sensor Fusion for Image-based Floorplan Construction. Project Tango [1] reconstructs 3D indoor structure in real-time by integrating extra depth sensing cameras and motion capture sensors, which significantly mitigates the computation and amount of images required for 3D modelling. Jigsaw [6] brings 2D indoor floorplan construction to commodity smartphones. Relative geometry of landmarks is derived by SfM and vanishing line detection, while mapping into the physical space is achieved by carefully designed 'Click-Walk-Click' micro tasks. IndoorCrowd2D [3] jointly fusion images captured by back camera and inertial sensory measurements along walking trajectory to construct building interior skeleton. *ClickLoc* is inspired by this thread of research that utilizes inertial sensors to correlate image-generated relative models to physical coordinates. Instead of constructing the whole building structure, *ClickLoc* aims to reduce the overhead of image database construction for localization. Therefore *ClickLoc* focuses on discrete POIs and tries to eliminate obtrusive interactions (*e.g.*'Click-Walk-Click' [6]).

Sensor Fusion for Image-based Location-based Services. OPS [18] combines GPS, inertial sensors and photos of the same object from multiple views to provide an outdoor object localization service. CamLoc [26] further reduces the effort into two photos at the same angle during arm stretching. iSee [21] aggregates direction-annotated images from multiple users to cluster the locations of the events in the photos. *ClickLoc* adopts the same principle to fusion inertial measurements and photos, but targets at localizing the user (camera) rather than the objects in the photo. iMoon [4] provides an image-based indoor navigation service by integrating photo-generated mesh with inertial-recognized user trajectories. While iMoon also enables user localization, the 3D navigation mesh is constructed relying on images and floorplan data only. Thus it requires orders more photos to bootstrap. Complimentarily, *ClickLoc* provides accurate user localization while reducing the overhead of image database construction, which can be integrated with [4]. Sextant [30] extracts the direction of pre-deployed physical features (images) from inertial sensors and localizes users by triangulation with at

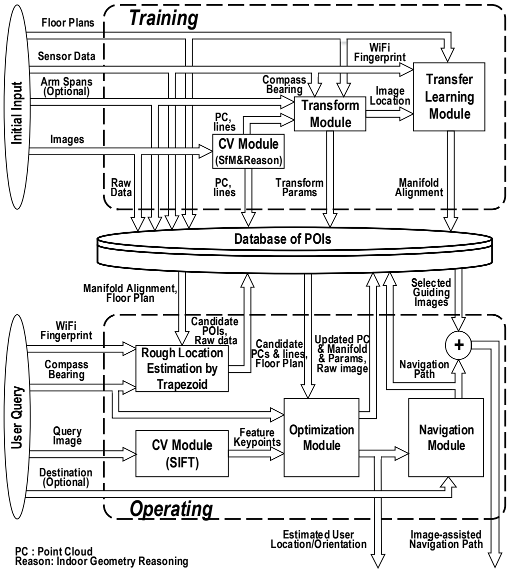


Figure 1. System overview of *ClickLoc*.

least three photos. *ClickLoc* advances Sextant by localizing users using a single photo.

SYSTEM OVERVIEW

Workflow from the user's perspective

When a user activates *ClickLoc*, the phone camera is automatically turned on, along with the WiFi and MEMS sensors (accelerometer, gyroscope, and magnetometer for device attitude estimation, denoted as compass for simplicity). Then the user takes a photo of a nearby POI at the center of his/her viewfinder. Afterwards, *ClickLoc* will send the photo and the sensor data to the *ClickLoc* server and a location tag will be returned and displayed to the user.

Workflow from the server's perspective

In the training stage (Figure 1), crowdsourced photos are input to the CV modules to extract visual features of POIs. Meanwhile, sensor readings (WiFi, compass bearings, and arm spans) are used to extract geometrical parameters. Both images and sensor data, together with the floorplan, constitute the database of POIs.

In the operating stage, a user sends a location query (including image, compass bearing, and WiFi) to *ClickLoc* server. Firstly, a rough location is estimated by Manifold Alignment [8] and trapezoid representation to reduce the search space of image matching. Then *ClickLoc* queries the database for the candidate point clouds (w.r.t. candidate POIs), and the correct POI will be identified by a fast image matching algorithm [24]. Finally the Optimization Module fusions vision and sensor data and outputs a fine-grained location estimate. Additionally, the Navigation Module can compute the route based on user's location queries and the destination.

SYSTEM ESTABLISHMENT AND UPDATE

ClickLoc relies on a database of POIs to operate. This section presents how *ClickLoc* constructs and updates the database.

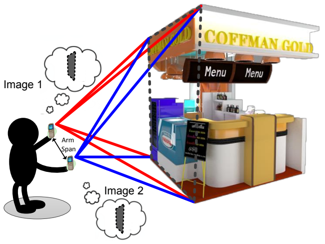


Figure 2. Illustration of how to add a POI into database.

Database Establishment

ClickLoc does not require dedicated user calibration to register a new POI into the database. For POIs with the desired properties (on the floorplan and has flat facade), *ClickLoc* allows users to take photos from any position, as long as the majority of the POI facade is included in the viewfinder. For POIs without the desired properties (*e.g.* a sculpture which does not appear on the floorplan, or a shop with round facade), *ClickLoc* instructs the user to stretch his/her left arm to take a photo, and take another photo with his/her straightened right arm (Figure 2). As few as two photos (and the distance of arm span) are adequate in theory, while the more photos, the better localization accuracy. Lastly, *ClickLoc* asks the user to pinpoint the POI on the digital floorplan. Given the above information, a POI can be registered.

Database Update

Usually hundreds of overlapping images are needed for SfM to compute an accurate dense point cloud [6, 28]. However, *ClickLoc* uses only a few images to bootstrap and can evolve by learning from user queries. Although *ClickLoc* might be inaccurate with only a few images, it is sufficient to identify the POI and estimate a rough location. Then the query image can be absorbed into the 3D model of the POI, such that, a more accurate point cloud can be generated (Detailed in System Establishment Evaluation).

FROM 3D POINT CLOUD TO PHYSICAL LOCATION

This section describes solutions to mapping the image-generated relative 3D point cloud to the physical location.

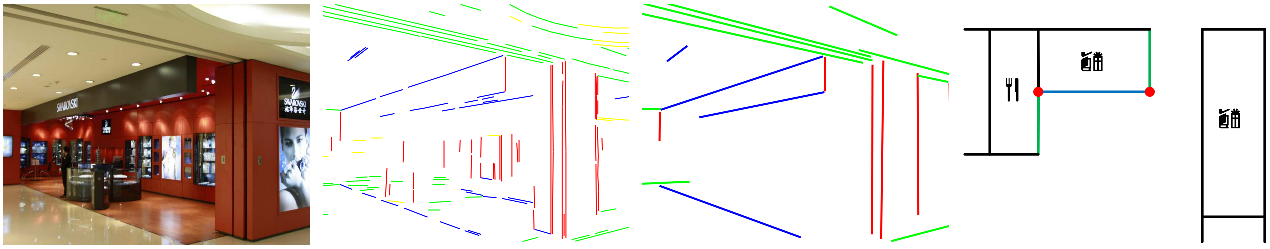
Preliminaries

Structure from Motion.

SfM is a classic CV technology to derive a 3D model of objects in the visible scene [13]. Its input is multiple photos of an object from different locations and orientations (or from different cameras). For each photo, SfM runs a feature detection algorithm (*e.g.*, SIFT [17]) and identifies various keypoints. By matching multiple sets of keypoints, SfM reconstructs: 1) a sparse 3D point cloud of the geometry captured by those keypoints; 2) the relative positions and orientations of the cameras when the original photographs were taken [13]. However, pure SfM has two limitations: 1) Hundreds of overlapping images are required for SfM to compute an accurate dense point cloud [6, 28]; 2) Unknown relative scaling, rotation, and translation requires a separate solution to transform the image space into the physical space [18].

Indoor Geometric Reasoning.

Indoor Geometric Reasoning [14] assumes that indoor environments satisfy the Manhattan World assumption. Specifically, most planes lie in one of three mutually orthogonal



(a) Original Image (b) Extracted lines (c) After Processing (d) Map to Floor Plan

Figure 3. Indoor Geometric Reasoning by Line Segments

orientations. In addition, indoor environments usually have a floor plane and a single ceiling plane with a constant height. Under these assumptions, projections of buildings are geometrically constrained by a small set of rules defined on the connection of walls. Then the orientations of line segments can be determined by the building hypotheses (Figure 3b).

Estimation of Scaling

Indoor Geometric Reasoning, Utilizing Floor Plan

The scaling factor λ proportionally shrinks or expands the point-to-point distances in the SfM point cloud to match the corresponding real world distances in meters. Given the distance between two points in both the image and physical spaces, we can obtain λ by division. To find such correspondence, *ClickLoc* estimates λ by extracting the width of the POI's facade using Indoor Geometric Reasoning [14].

However, the line segments generated by [14] are intermittent due to clutter of various objects in complex indoor scenarios (Figure 3b), and many are inconsistent with the POI's contour. Thus we first merge or filter the short line segments via a threshold (Figure 3c). Then the height of the POI equals the distance between the ceiling plane and the floor plane, which is the distance between the longest parallel lines in the image. Finally, the length of these parallel lines are regarded as the width of the POI. By projecting the parallels back into the 3D model, we can obtain their end point coordinates in the image space [6] and physical space (Figure 3d). We assume the physical width of a POI is known via the floorplan.

Minimization of Arm Span Error, Considering Sensor Data

The above scaling estimation relies on the floorplan, and it assumes that the POI has a flat facade or planar shape. To estimate λ of POIs without these desired properties, *ClickLoc* utilizes the distance between each pair of images. Ideally, the distance between each pair of images is similar to the user's arm span (Figure 2). However, in practice, the user may not know his/her arm span. And it is often difficult to straighten the arm while taking photos. To overcome these problems, we first minimize the arm span error using vision information. Let $(L_i^x, L_i^y)/(R_i^x, R_i^y)$ be the coordinates of left/right camera centers for the i^{th} pair of photos in point cloud (obtained by SfM). Then the arm span s equals the distances between any pair of camera centers in vision coordinates $\sqrt{(L_i^x - R_i^x)^2 + (L_i^y - R_i^y)^2}$ after multiplying the vision distance by the scaling factor λ. Since the arm span may contain error relative to each pair of images, we introduce error terms for the arm span E_i^s. Then the optimization is to minimize on the sum of $|E_i^s|$ as follows:

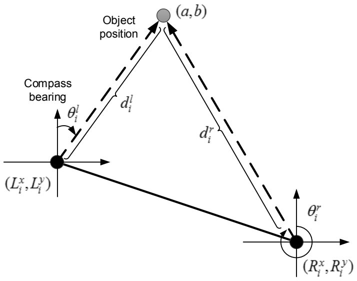


Figure 4. Illustration of compass-based trilateration.

$$\textbf{Minimize} \quad \sum_{\forall i} |E_i^s| \tag{1}$$

$$\textbf{Subject to}$$

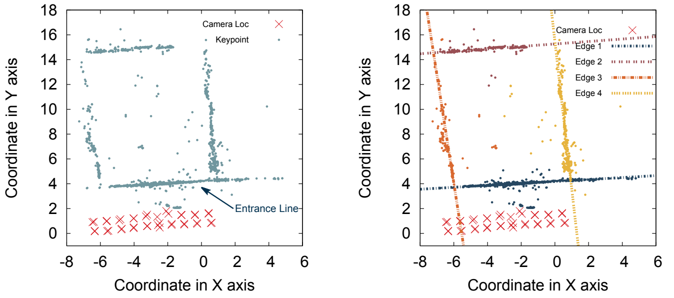
$$\forall i \quad : \quad |s + E_i^s| = \lambda\sqrt{(L_i^x - R_i^x)^2 + (L_i^y - R_i^y)^2}$$

To reduce the impact of human errors, we introduce another mechanism to calibrate λ using the compass. The rationale is that, an alternative type of compass-based trilateration can be achieved as in Figure 4. Specifically, let d_i^l/d_i^r be the real world distance between left/right camera center $(L_i^x, L_i^y)\,/\,(R_i^x, R_i^y)$ and the object position (a, b). Let θ_i^l/θ_i^r be the compass bearing for left/right camera. If we regard the object position as the origin, the coordinates of left and right camera centers can be represented by $(-sin\theta_i^l d_i^l, -cos\theta_i^l d_i^l)$ and $(-sin\theta_i^r d_i^r, -cos\theta_i^r d_i^r)$. The arm span s_i obtained from the previous optimization then equals the distance between the two camera centers $s_i = \sqrt{\left(-sin\theta_i^l d_i^l + sin\theta_i^r d_i^r\right)^2 + \left(-cos\theta_i^l d_i^l + cos\theta_i^r d_i^r\right)^2}$. Considering compass noise, we introduce error terms E_i^l and E_i^r for each pair of images. With these error terms, we aim to find λ that minimizes the sum of compass bearing error $\left|E_i^l + E_i^r\right|$. The whole optimization is as follows:

$$\textbf{Minimize} \quad \sum_{\forall i} \left|E_i^l + E_i^r\right| \tag{2}$$

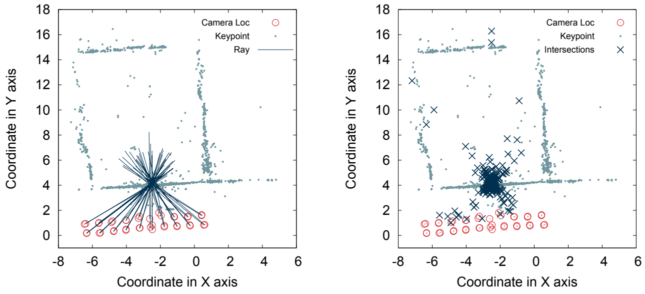
$$\textbf{Subject to}$$

$$\begin{aligned}
\forall i \quad &: \quad s_i^2 = \left[-sin\left(\theta_i^l + E_i^l\right) d_i^l + sin\left(\theta_i^r + E_i^r\right) d_i^r\right]^2 \\
&\qquad + \left[-cos\left(\theta_i^l + E_i^l\right) d_i^l + cos\left(\theta_i^r + E_i^r\right) d_i^r\right]^2 \\
\forall i \quad &: \quad s_i = s + E_i^s \\
\forall i \quad &: \quad d_i^l = \lambda\sqrt{(L_i^x - a)^2 + (L_i^y - b)^2} \\
\forall i \quad &: \quad d_i^r = \lambda\sqrt{(R_i^x - a)^2 + (R_i^y - b)^2}
\end{aligned}$$



(a) Point cloud and cameras (b) 4 edges found by “K-Edges”

Figure 5. Illustration of the top view of a point cloud



(a) Rays emit from camera centers (b) Intersections of Views

Figure 6. An illustration of the intersection of Views

It is worth mentioning that although compass readings are noisy and may have non-zero bias due to indoor magnetic interferences [23], we dramatically mitigate the error. The first constraint can be transformed into the following equation,

$$s_i^2 = {d_i^l}^2 + {d_i^r}^2 - 2cos\left(\theta_i^l - \theta_i^r + E_i^l - E_i^r\right) d_i^l d_i^r \qquad (3)$$

Since the compass bearing differences are used instead of the absolute bearings, the inherent bias can be eliminated.

Estimation of Rotation

Find the Entrance, Extended Vision Method

The result of Indoor Geometric Reasoning further contributes to our rotation estimation solution. If we look at the top view of the 3D point cloud as in Figure 5a, the facade of a POI forms a dense line and we denote it as an entrance line. Furthermore, the projection of the longest pair of parallels obtained by scaling estimation naturally coincide with the entrance line. Once the slope of the entrance line in the 3D point cloud is calculated, the rotation can be determined by referring to the floorplan. Not only the entrance lines, but also the wall lines on the top view can be identified and reconstructed (*e.g.*, Figure 5b). We also propose “K-Edges” to find them in the point cloud. Similar to the classical k-means algorithm, “K-Edges” uses an iterative refinement technique. Given a point cloud with n feature points $(f_1, f_2, ..., f_n)$ and an initial set of k lines $\{a_i^{(1)}x + b_i^{(1)}y + c_i^{(1)} = 0, i \in [1, k]\}$ from Indoor Geometry Reasoning (by identifying and filtering line segments on the floor plane, similar to Figure 3c), and the number of k can be estimated by referring to the floorplan (typically POIs are polygons on the floorplan). Then it runs by alternating between the Assignment Step (Equation 4) and the Update Step (Equation 5) as in Algorithm 1.

$$S_i^{(t)} = \{f_p : \left|\frac{a_i^{(t)}x_p + b_i^{(t)}y_p + c_i^{(t)}}{{a_i^{(t)}}^2 + {b_i^{(t)}}^2}\right| \le \left|\frac{a_j^{(t)}x_p + b_j^{(t)}y_p + c_j^{(t)}}{{a_j^{(t)}}^2 + {b_j^{(t)}}^2}\right| \forall j, 1 \le j \le k\} \qquad (4)$$

where the p^{th} feature point has $f_p = (x_p, y_p)$, and is assigned to exactly one line.

$$(a_i^{(t+1)}, b_i^{(t+1)}, c_i^{(t+1)}) = LeastSquareFitting(S_i^{(t)}) \qquad (5)$$

The algorithm converges when the assignments no longer change. The line constructed by the most feature points is assumed to be the entrance line, because it is the facade of the POI users are shooting for and it has the most images.

Algorithm 1: Heuristic K-Edges

```
Input  : Point cloud from a given POI, $(f_1, f_2, ..., f_n)$
         Initial lines,$\{a_i^{(1)}x + b_i^{(1)}y + c_i^{(1)} = 0, i \in [1, k]\}$
Output : Corresponding entrance line for the POI
Iteration t = 1
while $t \le$ Maximum number of iterations do
    Assignment Step:
    for each feature point $f_i$ in $(f_1, f_2, ..., f_n)$ do
        Assign $f_i$ to its nearest line. The point to line distance is
        measured in Euclidean distance.
    for each line $a_i^{(t)}x + b_i^{(t)}y + c_i^{(t)} = 0$ do
        Obtain its feature point set, ${S_i}^{(t)}$. (see Eqn. 4)
    Update Step:
    for each feature point set ${S_i}^{(t)}$ do
        Calculate its new fitting line. (see Eqn. 5)
    if $\left\{\left({a_i}^{(t)}, {b_i}^{(t)}, {c_i}^{(t)}\right), i \in [1, k]\right\} =$
    $\left\{\left({a_i}^{(t+1)}, {b_i}^{(t+1)}, {c_i}^{(t+1)}\right), i \in [1, k]\right\}$ then
        End while
    t = t + 1
Return the line $\left({a_i}^{(t)}, {b_i}^{(t)}, {c_i}^{(t)}\right)$ with largest $\left|{S_i}^{(t)}\right|$
```

Utilize Compass Bearing, Amortized Method

To estimate the rotation of POIs without flat facade or existence on the floorplan, we utilize the noisy compass bearing with a device attitude estimation algorithm A^3 [40] (yielding a 90-percentile error of $10°$). Then for each image, it has a compass bearing θ_i^c in physical world and an orientation θ_i^v in the 3D point cloud. Ideally, the difference between these two angles $|\theta_i^c - \theta_i^v|$ is the same among all images for the same POI (denoted by constant γ). By introducing error term E_i^θ relative to each compass reading, the rotation can be estimated by minimizing the sum of compass bearing errors.

$$\textbf{Minimize} \quad \sum_{\forall i} \left|E_i^\theta\right| \qquad (6)$$

$$\textbf{Subject to}$$

$$\forall i \quad : \quad \theta_i^c + E_i^\theta - \theta_i^v = \gamma$$

Estimation of Translation

Find the Corners, Extended Vision Method

The key of estimating a translation is to find a point with known locations in both the point cloud and the physical world. Here we exploit the corners of a POI (*e.g.*, Figure 5b).

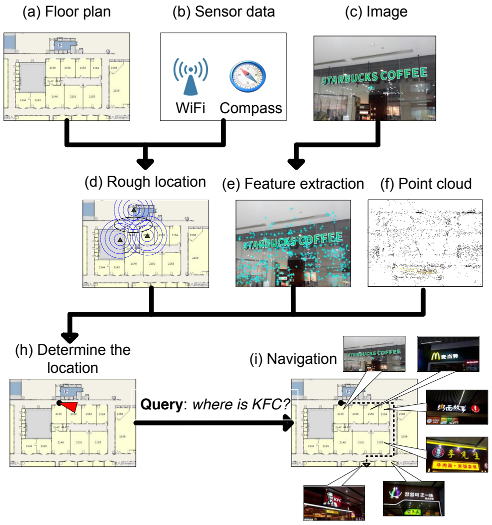


Figure 7. Procedure for Indoor Localization and Navigation

We first obtain the entrance line and wall lines of the POI via "K-Edges". Then the corners can be found by intersecting the entrance line with two wall lines. Finally we establish the mapping between the corners in the top view and the corners in the floor plan (*e.g.*, Figure 3d).

Find the Intersection of Views, DBSCAN-based Method

For POIs without the desired properties, we identify a reference point by analyzing user behavior. When a user takes a photo, he/she usually puts the POI at the viewfinder center. If we regard an image as a ray from the camera's center with the same orientation as the camera, the ray will traverse the POI (Figure 6a). Given n images for one POI, we have n rays from different locations. These rays result in up to $\binom{n}{2}$ intersections (Figure 6b). The intersections are densely distributed around the object, and DBSCAN [5] can be used. Once the densest cluster is obtained, its center can be treated as the reference point in the image space. The reference point in physical space is assumed to be the center of the facade for POIs without planar shape. For POIs that do not appear on the floorplan, the location pinpointed by the users is selected.

MULTI-MODAL LOCALIZATION AND NAVIGATION

This section details *ClickLoc*'s multi-modal localization and image-assisted navigation mechanism (Figure 7).

Multi-modal Localization

Pure CV-based localization could be both time and energy consuming, which leads us to the following solution.

Rough Location Estimation by Manifold Alignment

ClickLoc only possesses fingerprints uploaded together with query images. These fingerprints mainly concentrate on areas near POIs and distribute highly unevenly throughout the floorplan. We solve the above problem via Manifold Alignment [8] utilizing two observations: one is that the neighbouring grids in the physical space are likely to have similar WiFi fingerprints in signal space as well; the other one is that *ClickLoc* localizes users with considerable accuracy (detailed in the experiments), and the WiFi fingerprints uploaded with query images can be regarded as calibrated data to some extent. Then we divide the floor plan into grids, of which the grids' physical coordinates consist of the source dataset. The corresponding WiFi fingerprints of the grids consist of the target dataset. The above two observations provide **Strong neighborhood Correlation** and **Common Lower Dimensional Correlation** among source dataset and target dataset, which builds the theoretical basis of Manifold Alignment [8]. Then it further learns the mappings between two datasets characterized by the same underlying manifold via a dimensionality reduction based transfer learning scheme.

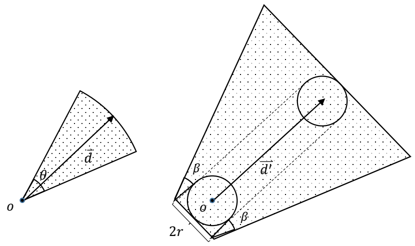


Figure 8. Trapezoid Representation of an Image.

To preserve the neighborhood correlation, we adopt the same locally linear embedding technique in [29]. For each higher dimensional data point p_i, the N data points having the smallest distances to it form its neighbor set $\mathbb{N}(i)$. Let $\left[p_i^{\mathbb{N}(1)}, ..., p_i^{\mathbb{N}(N)}\right]$ be its set of N neighbouring data points. The neighborhood weights of p_i are calculated by:

$$\begin{aligned} \textbf{Minimize} \quad & \left| p_i - \sum_{j\in\mathbb{N}(i)} W_{ij} p_i^{\mathbb{N}(j)} \right| \\ \textbf{Subject to} \quad & \sum_{j\in\mathbb{N}(i)} W_{ij} = 1 \end{aligned} \tag{7}$$

where W_{ij} is the neighborhood weight of data point p_j to p_i. Consider the source dataset X and the target dataset Y, with $X_i \in \mathbb{R}^m$ and $Y_i \in \mathbb{R}^n$. The projection functions from two higher spaces to the lower space are denoted by ϕ_X and ϕ_Y. We also have an indicative function I, where $I_{ij} = 1$ if the j^{th} fingerprint is in the i^{th} grid or $I_{ij} = 0$ otherwise. Then the loss function of this manifold alignment problem is:

$$\begin{aligned} \arg\min_{\phi_X,\phi_Y} \quad & \{ \mu \sum_{i,j} \|\phi_X(X_i) - \phi_X(X_j)\|^2 W_{ij}^X + \\ & \mu \sum_{i,j} \|\phi_Y(Y_i) - \phi_Y(Y_j)\|^2 W_{ij}^Y + \\ & (1-\mu) \sum_{i,j} \|\phi_X(X_i) - \phi_Y(Y_j)\|^2 I_{ij} \} \end{aligned} \tag{8}$$

Minimizing the first term guarantees that the larger W_{ij}^X, the smaller the $\|\phi_X(X_i) - \phi_X(X_j)\|$, which maintains the neighborhood relations of X within the elements of ϕ_X. The rationale is the same for the second term. The third term penalizes the discrepancies among correspondences, and μ is used to adjust the weight of different components.

Then during rough localization, when a query WiFi fingerprint comes, we map it into a low-dimensional space via Manifold Alignment and return its nearest neighbor in that space. The corresponding physical location of the nearest neighbor will be regarded as the rough location estimation.

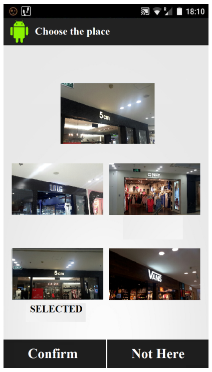


Figure 9. Match Correction

#POIs	Mall	Station
Top 1	93.4%	90.5%
Top 2	95.3%	93.7%
Top 3	97.8%	96.5%
Top 4	98.4%	97.6%
Top 5	98.9%	98.1%

Table 1. Image Match Accuracy

Candidate POI Selection by Trapezoid Overlapping

After obtaining the rough location estimation of a query image, we select candidate POIs by checking the overlapped area between the image's trapezoid representation (Figure 8) and the floorplan. Ideally, if we take a photo from O, the effective area covered by the photo can be represented by a sector (Figure 8), where θ is the field-of-view of the camera lens, $\vec{d}$ is the orientation of the camera when the photo is taken and $|\vec{d}|$ is the scene depth. However, the location estimation is coarse-grained, the orientation obtained from mobile phone compass is highly error-prone, and the actual scene depth is unknown. Thus we use a trapezoid to bound the effective area to account for these errors and uncertainties. Here r is the error of rough location estimation, $\beta = \theta/2 + \alpha$ with compass error α, and $|\vec{d'}|$ is the largest possible scene depth, where: $|\vec{d'}| \sim$ focal length $(mm) *$ POI height $(mm) *$ image height (pixels) / (POI height (pixels) $*$ CMOS height (mm)). Focal length can be obtained when taking the photo, POI height in pixels can be estimated by indoor geometry reasoning (distance between ceiling plane and floor plane), physical POI height can be determined by previously introduced Estimation of Scaling, while image height and CMOS height is known via smartphone. Then θ can be calculated correspondingly and any POI that overlaps with the trapezoid will be double checked by image match.

Image Matching

Motivated by [24], we implement a direct $2D - 3D$ matching framework using visual words for fast image matching. When a query image comes, *ClickLoc* conducts a simple threshold-based mechanism to judge whether the POI is correctly identified. If the image matching score exceeds the threshold, *ClickLoc* executes the next step directly. However, the photos could be in low quality (*e.g.*, unclear, blurred, and lose focus), or even not in our database. So we adopt an image match correction mechanism when the image matching score is below the threshold. As illustrated in Figure 9, *ClickLoc* retrieves the top 4 candidate POIs and ask the user to identify the correct one by tapping the thumbnail images. As summarized in Table 1, the accuracy improves from 93.4% to 98.4% in the mall and from 90.5% to 97.6% in the station by considering the top 4 candidates (dataset will be detailed later).

Image-assisted Navigation

ClickLoc navigates users by offering instructive images along the way. Since the crowd-sensed images are from different orientations, *ClickLoc* can provide the images with the orientation the user is going to face. We implement a simple image-assisted navigation mechanism. When a user types his/her query destination, *ClickLoc* returns the shortest path from his/her current location to the target location. Then we divide the path into segments according to its length. For each segment, we pick a POI with the most images, and the image with the closest orientation towards the user's coming direction is selected as a guiding image (Figure 7i).

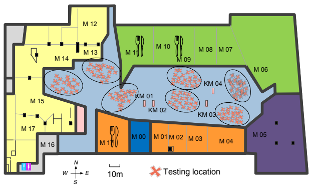


Figure 10. Shopping Mall.

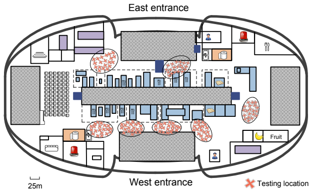


Figure 11. Railway Station

PERFORMANCE EVALUATION

Experimental Settings

Implementation

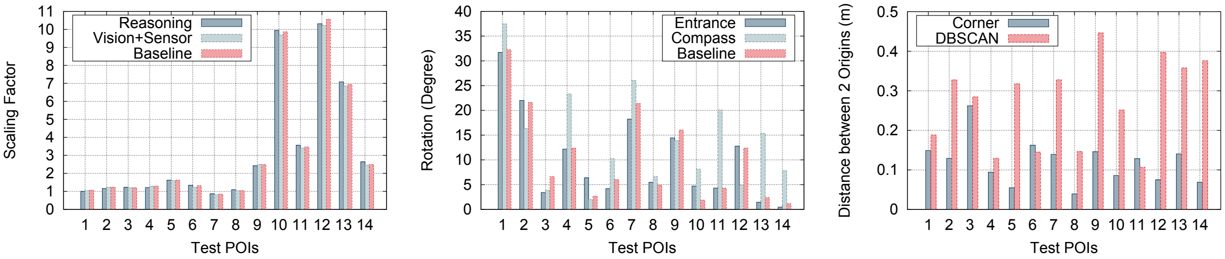
We prototype the *ClickLoc* front-end on a Google Nexus 5 phone and a Samsung Note 10.1 tablet, as a Java extension to the standard Android 4.4 camera program to ensure the validity and promptness of the sensor data. Photos and sensor data (accelerometer, gyroscope, magnetometer, and WiFi) are uploaded to a MacBook Pro running Mac OS X 10.9.2 as server. We use SIFT [17] for image feature keypoint extraction and Bundler [28] for SfM. We also use VisualSFM [32] to validate and visualize our results.

Data Collection.

We evaluate *ClickLoc* in a shopping mall (about $5,200m^2$ accessible area, denoted by Mall) and a railway station (about $23,700m^2$ accessible area, denoted by Station). In each scenario, we choose 7 POIs and more than 300 test locations. The number of photos taken for a POI range from 40 to 120, and more than 1000 photos are taken. The test locations are scattered as in Figure 10 and Figure 11. Two volunteers take 100 photos by randomly walking and taking photos of potential POIs inside the mall and station. The ground truth locations are recorded manually. We also evaluate *ClickLoc* on another dataset from Argus [33], which includes a shopping mall (50 POIs, denoted by Mall*) and a food plaza (41 POIs, denoted by Plaza), of which $5 \sim 20$ photos are taken for each POI and 200 photos from volunteers.

Performance of Transformation

To evaluate our transformation mechanism, we use Absolute Orientation [10] as the baseline. That is, given the test locations' coordinates in the 3D point cloud and their ground truth locations in the physical world, Absolute Orientation gives us an optimal transformation between the two coordinate systems. In other words, knowing the ground truth, Absolute Orientation outputs the most accurate scaling factor λ, rotation matrix R, and transformation vector. All the 14 POIs (Mall and Station) have flat facades and we test different transformation estimation methods on them for evaluation.



(a) Evaluation of Scaling (b) Evaluation of Rotation (c) Evaluation of Translation

Figure 12. Evaluation of Transformation

Evaluation of Scaling

We propose two methods to estimate λ. One estimates the width of a POI by Indoor Geometry Reasoning (denoted as Reasoning). The other minimizes the arm span error using vision coordinate constraints and compass bearing differences (denoted as Vision+Sensor). According to Figure 12a, both methods estimate λ precisely with an average error smaller than 3%. This result demonstrates the effectiveness of Indoor Geometry Reasoning. Further, we can estimate λ with dedicated human participation (taking photos in pairs), which enables scaling estimation for POIs without desired properties. Since the performance of Vision+Sensor is comparable to Reasoning, λ can thus be estimated accurately by jointly minimizing human errors and sensor errors. The above evaluation is based on 20 random images (or 10 pairs detailed in System Establishment) for each POI.

Evaluation of Rotation

Two methods are proposed to estimate the rotation. The first finds the entrance line in the top view of point cloud (denoted as Entrance) and the other calculates the rotation by minimizing the compass error (denoted as Compass). As shown in Figure 12b, Entrance is considerably accurate by aligning the entrance line with the floor plan with an average error of less than two degrees. However, results are disappointing for Compass due to the error-prone compass bearings, even we have adopt the newest strategy proposed in [40]. Here we make two notes of "K-Edges" algorithm in practice: 1) While K can be estimated accurately from floor plan, the K initial edges are suggested to be parallel or orthogonal. 2) The fitted edges may not strictly coincide with the physical walls due to feature points inside the POI. So we further filter out the noise and distant points when running the algorithm.

Evaluation of Translation

We propose two methods for translation estimation. The first finds the corresponding corners in image space and the floorplan (denoted as Corner). The second uses DBSCAN to find the densest cluster center of view intersections (denoted as DBSCAN). When the origin in the point cloud coordinate system is projected to the physical world, we obtain a new coordinate. The closer the new coordinate generated by Corner and DBSCAN to the one generated by Absolute Orientation is, the more accurate the translation. Figure 12c illustrates their performance. Corner outperforms DBSCAN in most cases, since users may not put the target in the middle of the viewfinder. The average distance between the two origins projected by Corner is $12.78cm$, and $27.16cm$ for DBSCAN.

Table 2. Multi-modal Localization Accuracy and Efficiency

	Mall	Station	Mall*	Plaza
Rough Localization	$11.45m$	N/A	$5.37m$	$7.14m$
Candidate POI#	1.25	3.38	1.47	1.71
Image Mismatch	1.6%	2.4%	2.3%	4.2%
Localization Latency	$8.64s$	$13.82s$	$8.59s$	$9.58s$

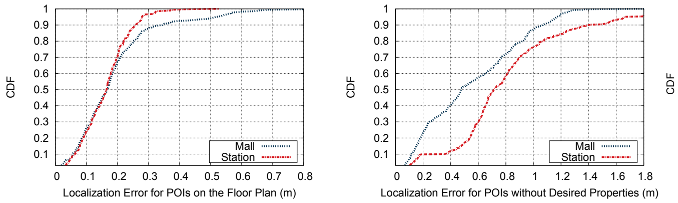
Accuracy of User Localization

Multi-modal Localization Accuracy and Efficiency

Table 2 summarizes the performance of our multi-modal strategy among four test locations. The experiments were conducted by using the photos taken for POIs as the training set (parameters were chosen optimally according to the training set). The photos taken by volunteers were regarded as the testing set. **Rough Localization** measures the average error of Manifold Alignment trained by sparse WiFi fingerprints and floorplan. The sparse WiFi fingerprints collected during image capture are enough for Manifold Alignment in the Mall, Mall*, and Plaza, while it is inapplicable for rough localization in the station due to the POI sparseness (15% coverage is suggested to provide comparable performance with full-fingerprinting localization [29]). The result also indicates that the rough localization accuracy increases with the more coverage with POIs. **Candidate POI#** measures the number of POIs needed to match before identifying a match or deciding the corresponding POI is not in the dataset. In the Mall, Mall*, and Plaza, the candidate POIs were selected by trapezoid overlapping as in Figure 8 with $r = 8m, \alpha = 20°$. In the station, we match candidate POIs in the descending order of their WiFi fingerprint similarity between the query image's WiFi fingerprint and the WiFi fingerprints for each POI. **Image Mismatch** shows the percentile of image match failure (not among the top 4 candidates, Figure 9). By checking the mismatch instances, 62.5% failures were caused by extreme distances or angles, 25% were caused by similar appearances and 12.5% was caused by obstructions. **Localization Latency** measures the time from taking the photo to showing the user's location on the phone screen. As we can see, the full adoption of our proposed multi-modal strategy roughly halve the candidate POI number and the localization latency.

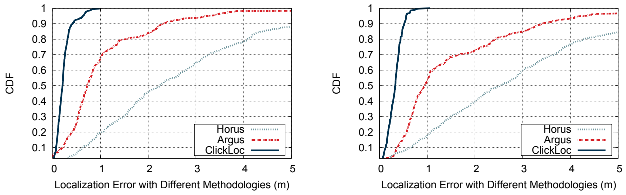
Overall Localization Performance

To measure the overall localization performance, we use half of the photos taken for POIs as the training set (Mall and Station) and the other half as the testing set (together with the matched photos from volunteers). We apply Reasoning, Entrance, and Corner methods to scaling, rotation, and



(a) POIs with Desired Properties (b) POIs without Desired Properties

Figure 13. Overall Localization Performance



(a) Mall* (b) Plaza

Figure 14. ClickLoc v.s. Argus

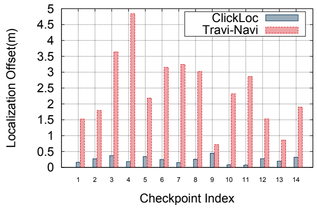


Figure 15. ClickLoc v.s. Travi-Navi

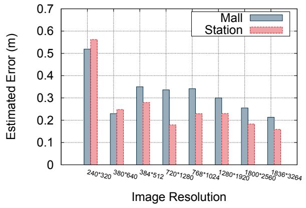


Figure 16. Impact of Photo Quality

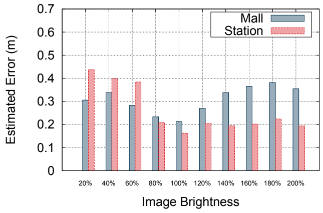


Figure 17. Impact of Illumination

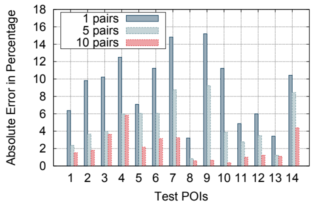


Figure 18. System Establishment

translation respectively for POIs on the floorplan. We also use Vision+Sensor, Compass, and DBSCAN for POIs without the desired properties. Figure 13 presents the overall localization performance. For POIs on the floorplan, the 50-percentile and 80-percentile errors are within $0.17m$ and $0.26m$. The 50-percentile and 80-percentile errors for POIs without the desired properties are about $0.7m$ and $1m$, which sheds lights on sub-meter level indoor localization. We then conduct two case studies to evaluate *ClickLoc* with two state-of-the-art of image-assisted localization and navigation.

Localization Case Study: ClickLoc v.s. Argus

Argus [33] is a recent work that enhances WiFi-based localization with visual clues by extracting geometric constraints from images and mapping the constraints jointly against the fingerprint space. We evaluate *ClickLoc*, Argus, and a classic RSS-based method Horus [36] on the same dataset [33]. As shown in Figure 14, *ClickLoc* significantly outperforms Argus and Horus in terms of localization accuracy, at the cost of a POI database instead of a WiFi fingerprint database.

Navigation Case Study: ClickLoc v.s. Travi-Navi

Travi-Navi[39] is a vision-assisted indoor navigation system that collects high quality images and inertial readings during a guider's walk on the navigation paths. The followers track the navigation trace, get visual instructions and tips, and receive alerts when they deviate from the correct paths. Differently, *ClickLoc* provides the navigation route on the floorplan with guiding images (Figure 7i). For comparison, we collect two navigation paths by traversing 7 POIs in Mall(Checkpoint $1 \sim 7$) and Station (Checkpoint $8 \sim 14$) respectively. We manually record the locations when the 3 followers reach the checkpoints along the recorded trace and ask the followers to take a photo of the POI for evaluation. The averaged location offsets estimated by *ClickLoc* and Travi-Navi among checkpoints are shown in Figure 15. Both methods successfully navigate the users to the destination. Travi-Navi provides satisfactory accuracy at the cost of collection of a guider navigation trace data, while *ClickLoc* provides more natural user interaction (displaying the route, and can start navigation anywhere) with the requirement of floorplan and POI database.

Influence Factors

Sensitivity to Quality of Photos

To emulate the impact of photo quality, we adjust the level of photographic detail by changing the photo resolution, and repeat the experiments for each resolution level. As shown in Figure 16, *ClickLoc* is more accurate with more photographic details, because SfM suffers from fewer keypoints.

Sensitivity to Illumination Change

To evaluate the performance of image match under different illumination conditions, we adjust the image brightness and contrast (since the lighting is stable in shopping malls and railway stations). As shown in Figure 17, *ClickLoc* is robust to small illumination change (80% and 120%) and is promising under illumination dynamics.

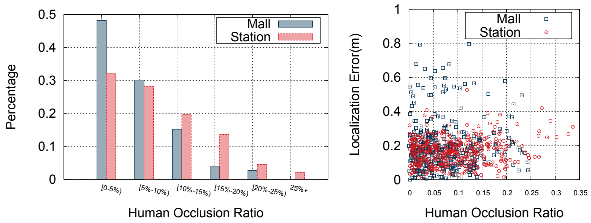
Sensitivity to Human Existence

The $1000+$ images for POIs and 200 images from volunteers in Mall and Station were taken without considering human existence, and some images may capture unwanted passengers. To evaluate the effect of human presence, we first apply Fast-RCNN [7] to estimate the ratio of human occlusion. Figure 19a shows the distribution of human occlusion among photos of two scenarios, and Figure 19b reveals the relationship between human occlusion ratio and localization error. We conclude that human existence does have a negative impact on the localization accuracy, since *ClickLoc* relies on image features to function. However, the decrease is acceptable as long as the majority of POI is clearly taken.

System Overhead

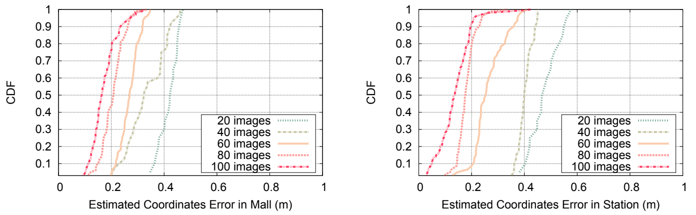
System Establishment

The user is required to take photos with the dedicated intention to add a POI into database. This is to estimate λ via Indoor Geometry Reasoning or arm span estimation. However, these operations are annoying, especially taking photos



(a) Human Occlusion Distribution (b) Localization Error

Figure 19. Human Occlusion Ratio and Impact on Localization



(a) Mall (b) Station

Figure 20. System Update Evaluation

in pairs. To measure the minimum pairs required, we estimate λ for each POI using the Vision+Sensor method with 1, 5, and 10 pairs. As in Figure 18, the estimated λ is closer to the Absolute Orientation estimation with more pairs of photos. During the experiments, we assume that 10 pairs of photos are enough for scaling factor estimation.

System Update

To emulate the procedure of system update, we choose the POIs with more than 100 photos in our database (3 POIs in Mall and 3 POIs in Station). Each POI has 20 images during the system establishment (denoted by 20 images). Then we randomly choose 20, 40, 60, and 80 additional photos for each POI (denoted by 40, 60, 80, and 100 images). These photos are treated as query images. As shown in Figure 20, more photos contribute to higher localization accuracy, while 40 to 60 photos are enough for satisfactory results.

Energy Overhead

We evaluate *ClickLoc*'s energy overhead with the tools and methodology proposed in [37]. We run *ClickLoc* on a Samsung Note 10.1 tablet, and measure energy consumption of running nothing, WiFi scanning only, and consecutively using *ClickLoc*. WiFi scanning frequency is set to $0.5Hz$, and the screen is kept on to prevent the hibernating mode. The power consumption for the first two scenarios are $565mW$ and $578mW$, respectively. Then we consecutively use *ClickLoc* in groups. Each group consists of 10 location queries. We measured 10 groups (100 queries). The average elapsed time for a group is $135s$ and a group consumes 102.5 Joules, or at the power of 759mW. So *ClickLoc* consumes $194mW$ additional power, which is only one third of the average power when the screen is on ($565mW$). Considering that users only activate *ClickLoc* when in need and most energy-consuming operations (*e.g.* image match and SfM) are left to server, such overhead has little impact on battery life.

DISCUSSION

Requirement of Floorplan. *ClickLoc* needs a floorplan to extract physical constraints and estimates the transformation from the image space to the physical space. We assume floorplans are available from indoor localization service providers. In case they are unavailable, we leverage the research trend in crowdsourced automatic floor plan construction [27, 6, 3].

Cold Start. *ClickLoc* requires a database of POIs to operate. We assume the system establishment can be crowdsourced by staff members in places like shopping malls, museums, and airports. If an user encounters a POI that is not included in the database, it degenerates into displaying the indoor floor plan with narrowed area by rough location estimation. *ClickLoc* also invites the user to add the POI into our database and the registration operation is a one-time investment. Various incentive mechanisms [34, 12, 38] can be applied to attract customer to contribute their effort.

Obtrusive User Interaction. While *ClickLoc* enables one click indoor localization, it may involve obtrusive user interactions during database construction for POIs without the desired properties. *ClickLoc* invites users to take photos in pairs with stretched arms to estimate the scaling factor of the 3D model. Here we plan to apply some inertial sensing techniques [26] and more natural registration operation: taking photos from different locations (*e.g.*, a few steps away).

Physical Features. In *ClickLoc*, we assume that users understand which POIs are likely to be selected by the system or other users. During our evaluation, we select apparent physical features like logos, shop entrances where the practicality has been validated in [30, 33]. *ClickLoc* may suffer significant performance degradation when lacking imagery features (*i.e.*, boring rooms, hallways, and small offices), because it relies on such features to distinguish POIs and extract geometric constraints. When image match fails, *ClickLoc* degenerates to a Manifold Alignment based WiFi localization.

CONCLUSION

With the trends towards improved sensor accuracy, and adoption of the cloud for low-cost, scalable computation, we envisage widespread user-friendly and augmented-reality indoor location-based service. We propose *ClickLoc*, an easy-to-use image-based indoor localization system with multimodal sensing. *ClickLoc* enables a user to localize himself in sub-meter accuracy with a single click. Future directions include optimizing image processing to reduce data communication and storage overhead. We also plan to deploy *ClickLoc* at larger scale to evaluate its performance in more diverse and dynamic environments, as well as a more comprehensive comparison study with other sub-meter accuracy indoor localization systems.

ACKNOWLEDGEMENT

Han Xu and Ke Yi are supported by HKRGC under grants GRF-621413, GRF-16211614, and GRF-16200415, and by a Microsoft grant MRA14EG05. This work was also supported in part by the NSFC Major Program 61190110, NSFC under grant 61522110 and 61572366. This work partially was funded by NanoTera.ch with Swiss Confederation financing.

REFERENCES

1. `https://get.google.com/tango/`. Accessed: 2016-6-18.
2. Bonin-Font, F., Ortiz, A., and Oliver, G. Visual Navigation for Mobile Robots: A Survey. *Journal of Intelligent and Robotic Systems 53*, 3 (2008), 263–296.
3. Chen, S., Li, M., Ren, K., Fu, X., and Qiao, C. Rise of the Indoor Crowd: Reconstruction of Building Interior View via Mobile Crowdsourcing. In *Proc. of ACM SenSys* (2015), 59–71.
4. Dong, J., Xiao, Y., Noreikis, M., Ou, Z., and Ylä-Jääski, A. iMoon: Using Smartphones for Image-based Indoor Navigation. In *Proc. of ACM SenSys* (2015), 85–97.
5. Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proc. of ACM KDD* (1996), 226–231.
6. Gao, R., Zhao, M., Ye, T., Ye, F., Wang, Y., Bian, K., Wang, T., and Li, X. Jigsaw: Indoor Floor Plan Reconstruction via Mobile Crowdsensing. In *Proc. of ACM MobiCom* (2014), 249–260.
7. Girshick, R. Fast r-cnn. In *Proc. of IEEE ICCV* (2015), 1440–1448.
8. Ham, J., Lee, D., and Saul, L. Semi-supervised Alignment of Manifolds. In *Proc. of AISTATS* (2005), 120–127.
9. Harle, R. A Survey of Indoor Inertial Positioning Systems for Pedestrians. *IEEE Communications Surveys & Tutorials 15*, 3 (2013), 1281–1293.
10. Horn, B. K. Closed-form Solution of Absolute Orientation using Unit Quaternions. *Journal of the Optical Society of America A 4*, 4 (1987), 629–642.
11. Jiang, Y., Pan, X., Li, K., Lv, Q., Dick, R. P., Hannigan, M., and Shang, L. Ariel: Automatic Wi-Fi based Room Fingerprinting for Indoor Localization. In *Proc. of ACM UbiComp* (2012), 441–450.
12. Kawajiri, R., Shimosaka, M., and Kashima, H. Steered Crowdsensing: Incentive Design Towards Quality-oriented Place-centric Crowdsensing. In *Proc. of the ACM UbiComp* (2014), 691–701.
13. Koenderink, J. J., Van Doorn, A. J., et al. Affine Structure from Motion. *Journal of the Optical Society of America A 8*, 2 (1991), 377–385.
14. Lee, D. C., Hebert, M., and Kanade, T. Geometric Reasoning for Single Image Structure Recovery. In *Proc. of IEEE CVPR* (2009), 2136–2143.
15. Li, F., Zhao, C., Ding, G., Gong, J., Liu, C., and Zhao, F. A Reliable and Accurate Indoor Localization Method using Phone Inertial Sensors. In *Proc. of ACM UbiComp* (2012), 421–430.
16. Liu, H., Darabi, H., Banerjee, P., and Liu, J. Survey of Wireless Indoor Positioning Techniques and Systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 37*, 6 (2007), 1067–1080.
17. Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *Springer International Journal of Computer Vision 60*, 2 (2004), 91–110.
18. Manweiler, J. G., Jain, P., and Roy Choudhury, R. Satellites in Our Pockets: an Object Positioning System using Smartphones. In *Proc. of ACM MobiSys* (2012), 211–224.
19. Mautz, R., and Tilch, S. Survey of Optical Indoor Positioning Systems. In *Proc. of IPIN* (2011), 1–7.
20. Mulloni, A., Wagner, D., Barakonyi, I., and Schmalstieg, D. Indoor Positioning and Navigation with Camera Phones. *IEEE Pervasive Computing 8*, 2 (2009), 22–31.
21. Ouyang, R. W., Srivastava, A., Prabahar, P., Roy Choudhury, R., Addicott, M., and McClernon, F. J. If You See Something, Swipe Towards It: Crowdsourced Event Localization Using Smartphones. In *Proc. of the ACM UbiComp* (2013), 23–32.
22. Rai, A., Chintalapudi, K. K., Padmanabhan, V. N., and Sen, R. Zee: Zero-effort Crowdsourcing for Indoor Localization. In *Proc. of ACM MobiCom* (2012), 293–304.
23. Roy, N., Wang, H., and Choudhury, R. R. I am a Smartphone and I can Tell my Users Walking Direction. In *Proc. of ACM MobiSys* (2014), 329–342.
24. Sattler, T., Leibe, B., and Kobbelt, L. Fast Image-Based Localization using Direct 2D-to-3D Matching. In *Proc. of IEEE ICCV* (2011), 667–674.
25. Shangguan, L., Yang, Z., Liu, A. X., Zhou, Z., and Liu, Y. Relative Localization of RFID Tags using Spatial-Temporal Phase Profiling. In *Proc. of USENIX NSDI* (2015), 251–263.
26. Shangguan, L., Zhou, Z., Yang, Z., Liu, K., Li, Z., Zhao, X., and Liu, Y. Towards Accurate Object Localization with Smartphones. *IEEE Transactions on Parallel and Distributed Systems 25*, 10 (2014), 2731–2742.
27. Shen, G., Chen, Z., Zhang, P., Moscibroda, T., and Zhang, Y. Walkie-Markie: Indoor Pathway Mapping Made Easy. In *Proc. of USENIX NSDI* (2013), 85–98.
28. Snavely, N., Seitz, S. M., and Szeliski, R. Photo Tourism: Exploring Photo Collections in 3D. *ACM Transactions on Graphics 25*, 3 (2006), 835–846.
29. Sorour, S., Lostanlen, Y., and Valaee, S. Joint Indoor Localization and Radio Map Construction with Limited Deployment Load. *IEEE Transactions on Mobile Computing 14*, 5 (2015), 1031–1043.
30. Tian, Y., Gao, R., Bian, K., Ye, F., Wang, T., Wang, Y., and Li, X. Towards Ubiquitous Indoor Localization Service Leveraging Environmental Physical Features. In *Proc. of IEEE INFOCOM* (2014).

31. Tilch, S., and Mautz, R. Development of a New Laser-based, Optical Indoor Positioning System. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences Commission 1501* (2010), 575–580.

32. Wu, C. Towards Linear-Time Incremental Structure from Motion. In *Proc. of IEEE 3DV* (2013), 127–134.

33. Xu, H., Yang, Z., Zhou, Z., Shangguan, L., Yi, K., and Liu, Y. Enhancing Wifi-based Localization with Visual Clues. In *Proc. of ACM UbiComp* (2015), 963–974.

34. Yang, D., Xue, G., Fang, X., and Tang, J. Crowdsourcing to Smartphones: Incentive Mechanism Design for Mobile Phone Sensing. In *Proc. of ACM MobiCom* (2012), 173–184.

35. Yang, Z., Wu, C., and Liu, Y. Locating in Fingerprint Space: Wireless Indoor Localization with Little Human Intervention. In *Proc. of ACM MobiCom* (2012), 269–280.

36. Youssef, M., and Agrawala, A. The Horus WLAN Location Determination System. In *Proc. of ACM MobiSys* (2005), 205–218.

37. Zhang, L., Tiwana, B., Qian, Z., Wang, Z., Dick, R. P., Mao, Z. M., and Yang, L. Accurate Online Power Estimation and Automatic Battery Behavior based Power Model Generation for Smartphones. In *Proc. of IEEE/ACM/IFIP CODES+ISSS* (2010), 105–114.

38. Zhang, X., Yang, Z., Zhou, Z., Cai, H., Chen, L., and Li, X. Free Market of Crowdsourcing: Incentive Mechanism Design for Mobile Sensing. *IEEE Transactions on Parallel and Distributed Systems 25*, 12 (2014), 3190–3200.

39. Zheng, Y., Shen, G., Li, L., Zhao, C., Li, M., and Zhao, F. Travi-Navi: Self-deployable Indoor Navigation System. In *Proc. of ACM MobiCom* (2014), 471–482.

40. Zhou, P., Li, M., and Shen, G. Use It Free: Instantly Knowing Your Phone Attitude. In *Proc. of ACM MobiCom* (2014), 605–616.