Knitter: Fast, Resilient Single-User Indoor Floor Plan Construction

Ruipeng Gao*
School of Software Engineering
Beijing Jiaotong University, China
Email: rpgao@bjtu.edu.cn

Bing Zhou*, Fan Ye
ECE Department
Stony Brook University, USA
Email: {bing.zhou,fan.ye}@stonybrook.edu

Yizhou Wang
EECS School
Peking University, China
Email: yizhou.wang@pku.edu.cn

***Abstract*—Lacking of floor plans is a fundamental obstacle to ubiquitous indoor location-based services. Recent work have made significant progress to accuracy, but they largely rely on slow crowdsensing that may take weeks or even months to collect enough data. In this paper, we propose Knitter that can generate accurate floor maps by a single random user's one hour data collection efforts. Knitter extracts high quality floor layout information from single images, calibrates user trajectories and filters outliers. It uses a multi-hypothesis map fusion framework that updates landmark positions/orientations and accessible areas incrementally according to evidences from each measurement. Our experiments on 3 different large buildings and 30+ users show that Knitter produces correct map topology, and 90-percentile landmark location and orientation errors of $3 \sim 5m$ and $4 \sim 6^{\circ}$, comparable to the state-of-the-art at more than $20\times$ speed up: data collection can finish in about one hour even by a novice user trained just a few minutes.**

I. Introduction

Lacking of floor plans is a fundamental obstacle to ubiquitous location-based services (LBS) indoors. Recently some academic work have made admirable progress to automatic floor plan construction. They require only commodity mobile devices (e.g., smartphones) thus scalable construction can be achieved by crowdsensing data from many common users. Among others [16], [21], [25], [26], CrowdInside [4] uses mobility traces to derive the approximate shapes of accessible areas; realizing that inertial and WiFi data are inherently noisy thus difficult to produce precise and detailed maps, a recent work Jigsaw [14] further includes images to generate highly accurate floor plans.

Despite such progress, these approaches usually require large amounts of data, crowdsensed from many random users piece by piece, resulting in long data collection time (weeks or even months) before maps can be constructed. In this paper, we propose *Knitter*, which can construct complete, accurate floor plans within hours. Even in large complex environments such as shopping malls, the data collection for a level takes only about one man-hour's effort. Instead of crowdsensing the data from many random users, Knitter requires only one user to walk along a loop path inside the building to collect small amounts of measurement data. Knitter is highly resilient to low user skill and thus data quality: with just a few minutes' practice, a novice user can collect data that produce maps at quality on par to well trained users.

The greatly improved speed and resilience using sparse and noisy data are made possible by several novel techniques. A single image localization method extracts high quality relative spatial relationship and geometry attributes of indoor places of interests (POIs, such as store entrances in shopping malls, henceforth called *landmarks*). This greatly reduces the amount of data needed. Image-aided calibration and optimization-based cleaning methods correct noises in user trajectories, and align them on a common plane. Thus outliers causing significant skews are identified and filtered. Instead of making a single and final "best" guess of map layout [14], which becomes accurate only after large amounts of data, Knitter takes *multi-hypothesis* measurements. It accumulates measurement evidences upon each data sample, updates parallel possibilities of map layouts incrementally, and chooses those supported by the strongest evidences. Collectively these techniques enable Knitter to produce complete and accurate maps using sparse and noisy data from novice users. Specifically, we make the following contributions:

- We develop a novel localization method that can extract the user's relative distance and orientation to a landmark using a single image, and produce multiple hypotheses about the landmark's geometry attributes.
- We devise image-aided angle and stride length calibration methods to reduce errors in user trajectories, and optimization-based discrepancy minimization to align multiple trajectories along the same loop path, thus detecting and filtering outliers.
- We propose an incremental floor plan construction framework based on dynamic Bayesian networks, and design algorithms that update parallel map layout possibilities using evidences from measurement data, while tolerating inevitable residual noises and errors.
- We devise a landmark recognition algorithm that combines complementary data to determine measurement/landmark correspondence, and methods for accessible area confidence assignment under sparse data, neither fully addressed in previous work.
- We develop a prototype and conduct extensive experiments in three kinds of large (up to $140 \times 50m^2$), typical indoor environments: featureless offices and labs, and feature-rich shopping malls, with 30+ users. We find that Knitter achieves accuracy comparable to the state-of-the-art [14] (e.g., 90-percentile position/orientation errors at $3 \sim 5m$ and $4 \sim 6^{\circ}$), with more than $20\times$ speed up that costs only one hour's efforts of a single user, and the reconstructed map can be used directly for localization.

II. Overview

Knitter takes several components in system measurements, map fusion framework, and compartment estimation to pro-

*The first two authors contribute equally and this work is supported in part by NSF CNS 1513719 and China NSFC 61231010 and 61210005.

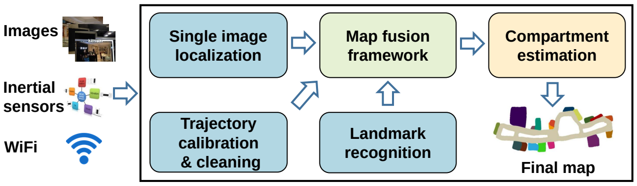


Fig. 1. Knitter contains several components to produce complete and accurate maps by a single random user's one hour data collection efforts.

duce the final map (shown in Figure 1) .

Three system measurement techniques are devised to produce inputs to the map fusion framework from sensing data: 1) *single image localization* extracts a landmark's geometry information, including its relative orientation, distance to the user, and its adjacent wall segment lengths from one image; 2) *trajectory calibration* leverages the image localization results to reduce user trajectory angle and stride length errors, then *trajectory cleaning* quantifies the trajectory quality and uses alignment and clustering to detect and filter outliers; 3) *landmark recognition* combines image, inertial and WiFi data of complementary strengths to determine which measurement data corresponds to which landmark, thus ensuring correct map update. The *map fusion framework* fuses previous measurement results to create maps under a dynamic Bayesian network formulation. It represents multiple possible map layouts each with different estimations of landmark positions as hidden states, represented by random variables, infers and updates their probability distributions incrementally, using evidences upon each additional measurement. The *compartment estimation* combines evidences from different kinds of measurements to properly assign accessible confidences to cells in an occupancy grid, such that estimations of compartment (e.g., hallways, rooms) shapes and sizes are accurate even with small amount of data.

III. Localization via a Single Image

Single image localization estimates the relative distance d and orientation θ of the user to a landmark in photo (shown in Figure 2). It also produces multiple hypotheses of the landmark's geometry attributes, with a weight (probability) for each hypothsis' measurement confidence. Such output is fed to the map fusion framework. Unlike most vision-based localization work [19] that relies on image matching to a database of known landmarks, we use line extraction and do not need any prior benchmark images.

Pre-processing. First we use Canny edge detector [5] to extract line segments (Figure 4(c)) from an image (Figure 4(a)). We cluster them [23] and find the vanishing point (VP) where the wall/ground boundary line and horizon line intersect, and obtain its pixel coordinates (u, v).

Estimating θ. Based on projective geometry, we can compute the relative orientation angle θ of the landmark to the camera using the vanishing point's coordinates:

$$\theta = \pi - mod(\arctan(\frac{u - \frac{W}{2}}{f}), \pi) \tag{1}$$

where W is the image width in pixels, f is the camera's focal length in pixels computed from the camera's parameter specifications.

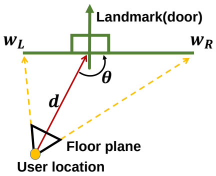


Fig. 2. Landmark's geometry layout.

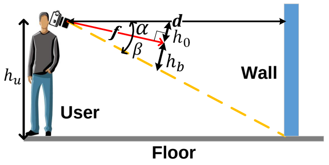
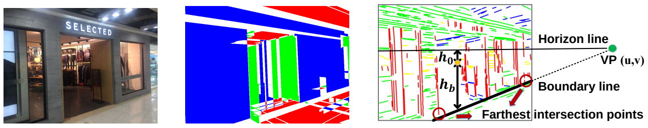


Fig. 3. Estimation of distance d.



(a) Example image (b) Orientation map (c) Boundary line

Fig. 4. Extracted horizon line and boundary line on the example image (better viewed in color). Red circles denote farthest intersection points between vertical line segments and boundary line.

Estimating d. Assuming the user points the camera downwards (or upwards) at an angle α (shown in Figure 3), d can be computed as:

$$\tan\alpha = \frac{h_0}{f}, \tan\beta = \frac{h_b}{f}, d = h_u \cdot \cot(\alpha + \beta) \tag{2}$$

where h_0 denotes the vertical distance of the horizon line to the image center, derivable from (u, v), h_b the vertical distance from the image center to the boundary line (both marked in Figure 4(c)), and h_u is the actual camera height which can be approximated using the user's height (input by the user or estimated).

Computing h_b in Equation 2 requires us to identify the floor-wall boundary line (Figure 4(c)). This is not straightforward because there may exist many other lines that are parallel to the true boundary. Reliably distinguishing them from the real one is difficult. Thus we develop a method that produces *multiple hypotheses* of floor-wall boundary so the correct one is included with high probability.

We first generate an orientation map [17] (Figure 4(b)) where the orientation of each surface is computed and its pixels colored accordingly. Given a floor-wall boundary candidate l_i, we compute the fraction of wall and floor pixels with consistent orientations as the weight:

$$w_{l_i} = \frac{S^+_{floor} + S^+_{wall}}{S^{all}_{floor} + S^{all}_{wall}} \tag{3}$$

where S^+_{floor} and S^+_{wall} denote the floor/wall pixel areas whose orientations conforming to l_i (i.e., above l_i are walls facing sidewards and below l_i are floors facing upwards), S^{all}_{floor} and S^{all}_{wall} the respective total pixel areas. The correct candidate should have the best consistency, thus greatest weight.

Estimating (w_L, w_R). Along a boundary line, we detect intersection points with vertical line segments. The left- and right-farthest intersection points are identified in Figure 4(c), and their horizontal pixel distances (w^p_L, w^p_R) to the image center are transformed into left and right wall segment lengths (w_L, w_R) based on projective geometry:

$$w_{L,R} = \frac{d \cdot \sin(\arctan(\frac{w^p_{L,R}}{f}))}{\sin(\theta \mp \arctan(\frac{w^p_{L,R}}{f}))} \tag{4}$$

Now we have multiple hypotheses, each having a boundary line, user distance/angle, and two wall segment lengths, with a

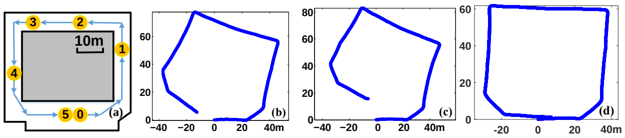


Fig. 5. Trajectories from (a) ground truth with 6 photo-takings; (b) gyroscope based [22], [29]; (c) phone attitude [30]; (d) image-aided angle calibration.

weight (probability). Detailed evaluations (Section VIII) show that this localization method generates quite small errors ($< 1m$) even at remote distances ($> 10m$).

IV. Trajectory Calibration and Cleaning

Accurate user trajectories from inertial data are critical in floor plan construction. In Knitter, the user walks along a closed loop path multiple times, taking landmark photos and collecting inertial data. Each loop may take about 10 minutes. Significant errors may accumulate during the long walk, and frequent stops to take landmark photos may create severe inertial disturbances, both resulting in deformed, inaccurate trajectories. We must be able to rectify such errors.

A. Trajectory Calibration

We tested two trajectory construction methods: a gyroscope based (Zee [22] and UnLoc [29]) and a recent phone attitude one (A^3 [30]). Although the step counts are relatively accurate, neither produces satisfactory trajectories due to walking direction errors. Figure 5(b) and 5(c) show their results for a 5-minute walk (Figure 5(a)). The main reasons are: 1) the gyroscope has significant drifts over long walking periods; 2) during long, straight walk, there are few calibration opportunities of similar changes in compass and gyroscope as required in A^3 [30]; 3) strong electromagnetic disturbances (e.g., server rooms [15]) can cause false “calibrations.” We propose image aided methods to calibrate the angles and stride lengths, thus accurate walking direction and trajectories (Figure 5(d)).

Image-aided Angle Calibration. Since gyroscopes are known to have linear drifts [30], we leverage “closed loops” to estimate an average gyroscope drift rate δ. After finishing a loop, the user returns to the starting area and takes a second photo of the first landmark. Using single image localization, we compute two angles θ_1, θ_2 based on Equation 1 for both images of that landmark. Their difference $\Delta\theta = \theta_1 - \theta_2$ is the orientation angle change. Since the user may not return perfectly to the starting point, this will cause an additional change in user orientation, which can be measured by the difference of the gyroscope's “yaw” between the two images, denoted as Δg. The rate δ and calibrated angle g_t^* are computed as:

$$\delta = \frac{\Delta g + \Delta\theta}{T}, g_t^* = g_t + \delta \cdot t \tag{5}$$

where T is the time between taking the two images. We find this method is not affected by electromagnetic disturbances; it always achieves accurate and robust angle calibration ($\sim 5°$ errors at 90-percentile).

Image-aided Stride Length Calibration. We leverage the closed loop to calibrate the stride length that may change in different regions, e.g. larger in wide and open hallways [4]. Our localization method can compute the user's relative location to the first landmark, thus the location change before and after the loop can be computed as a vector $\vec{v}$ pointing from the start to the end location. We compensate each point at time t on the trajectory with $\vec{v} \cdot t/T$ to calibrate stride length errors. Figure 6 shows that both angle and stride length calibrations are needed to produce an accurate closed loop trajectory (Figure 6(d)).

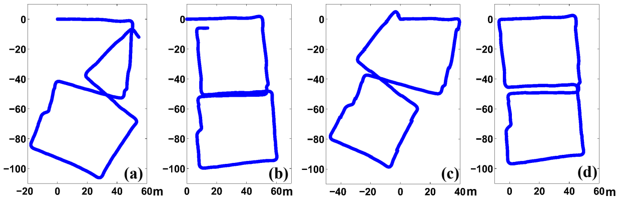


Fig. 6. (a) raw trajectory for a closed loop; (b) angle calibration only; (c) stride length calibration only; (d) both calibrations.

B. Trajectory Cleaning

Calibration only rectify trajectories with small errors, but not outliers. We conduct the following three steps to detect and filter out such outliers: loop screening, loop alignment, and outlier removal.

Loop Screening. We use the “gap”, the distance between the starting and ending locations of the angle-calibrated loop for preliminary screening. Since the user returns to the starting area, ideally the gap should be 0 after image compensation. A lower quality loop has a larger gap. Given multiple trajectories, we compute the standard deviation σ of the calibration shift vector's length $|\vec{v}|$ normalized over the size of the trajectory, and remove those with $|\vec{v}|$ beyond 3σ. [1]

Loop Alignment. Multiple trajectories must be placed within the same global coordinate system. However, the trajectories can not overlap perfectly with each other. Each time the exact path may differ slightly within the same hallways or isles, so do the stride lengths. Thus the trajectories have slightly different shapes and possibly scales.

Without loss of generality, we consider how to place a second trajectory with respect to an existing one. Initially, we pick the one with the smallest gap as a reference loop, and use landmark recognition (Section VI) to detect which landmark c_i on the second loop corresponds to landmark i on the reference loop. This addresses situations where the user takes photos of slightly different sets of landmarks in each loop (due to negligence or imperfect memory). Then we *translate, rotate and scale* the second one to achieve “maximum overlap” with the first one, as defined by minimizing the overall pairwise distances of corresponding landmarks:

$$\{\phi^*, O^*, s^*\} = \underset{\phi, O, s}{\operatorname{argmin}} \sum_{i=1}^{N} \| s \cdot R(\phi) \cdot (M_{c_i}^2 - O) - M_i^1 \|_2 \tag{6}$$

where $M_i^1 = X_i^1 + Z_i^1$ and $M_{c_i}^2 = X_{c_i}^2 + Z_{c_i}^2$ denote the coordinates of the i_{th} landmark in the reference loop and the corresponding landmark c_i in the second loop, X_i^1 and $X_{c_i}^2$ are the coordinates of photo taking locations of them, Z_i^1 and $Z_{c_i}^2$ are the relative locations from the user to the landmark (from single image localization). $\{\phi, O, s\}$ denote the rotation, translation and scale factors to the second trajectory, and $R(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$ is the rotation matrix. A simple greedy search for an initial solution followed by iterative perturbation can find the approximate solutions

[1] According to Chebyshev's Theorem, this removes those trajectories with extreme errors beyond 88.9% of all loops.

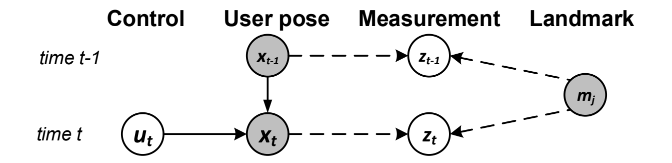


Fig. 7. Dynamic Bayesian Network. Gray nodes (user/landmark states) are hidden variables to be computed, and unshaded ones are observation variables measured directly. Arrow directions denote determining relationship, solid for movement update and dashed for landmark update.

for the three parameters. Each additional trajectory is placed similarly within the common coordinate system. [2]

Outlier Removal. After all trajectories and landmark sets are placed on the same coordinate system, we identify the common subset of s_m landmarks across all loops. We represent those in loop k with a multi-dimensional vector $(m^k_{s_1}, ..., m^k_{s_m})$, where $m^k_{s_i}$ is landmark s_i' location, and compute the Euclidian distance between each two vectors. Then we use a density-based clustering algorithm DBSCAN [10] to eliminate outlier loops: vectors are "reachable" to each other if the distance is within an empirically decided threshold $\varepsilon = 0.8m$, those not reachable from any other vector are detected as outliers, and respective loops removed.

V. Map Fusion Framework

With the image measurements in Section III and motion trajectory in Section IV, we need to estimate landmarks' positions and orientations in the global coordinate system. To this end, we use a Dynamic Bayesian Network framework to fuse the extracted information from previous measurement algorithms to build maps incrementally.

A. Dynamic Bayesian Network

We formally represent different states in the floor plan construction process as random variables, and denote their dependence using arrows (shown in Figure 7). We assume time is slotted. At each time t, x_t denotes the user pose (i.e., camera/phone coordinates and orientation); u_t is the control including the walking distance and heading direction that alter the user pose from x_{t-1} to x_t; z_t is measurement of the landmark by the user (e.g., relative distance d and angle θ); m_{c_t} are the coordinates and orientation of the landmark being measured, $c_t = j$ $(j = 1, ..., N)$ is the index of this landmark as detected by landmark recognition (Section VI).

In the above, u_t and z_t are *observation variables* that can be measured directly from sensors, while x_t and m_j are *hidden variables* that must be computed from observation ones. These variables are represented by probability distributions. Given control signal $u_{1:t}$ (shorthand for $u_1, ..., u_t$) and measurements $z_{1:t}$, the goal is to compute the posterior (i.e., conditional) probability of both landmark positions $m_{1:N}$ and user poses $x_{1:t}$, i.e. $p(x_{1:t}, m_{1:N}|u_{1:t}, z_{1:t})$.

B. Particle Filter Algorithm

We use a particle filter algorithm to compute the above user poses and landmark attributes incrementally. We maintain a collection of K "particles." Each particle k $(k = 1, ..., K)$ includes a different estimation of:

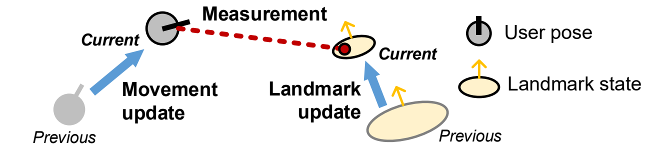


Fig. 8. A current user pose is computed based on the previous pose and control signal. Then a landmark's state is updated using a measurement from the new user pose.

- user pose x_t: user's coordinates (x, y) and heading direction φ,
- each landmark's mean μ and covariance Σ of its coordinates and orientation (μ_x, μ_y, μ_ϕ), assumed multivariate Gaussian distribution,
- two adjacent wall lengths (w_L, w_R) of each landmark.

At each time slot, we perform 5 steps to update the states in each particle k.

1. Movement Update: given the previous user pose x_{t-1} at time $t-1$ and recent control $u_t = (v, \omega)$ where v is the moving speed and ω the heading direction (obtained from trajectory measurement algorithms in Section IV), the destination is computed by dead reckoning. The current pose x_t is computed by picking a sample from a multivariate Gaussian distribution of many possible locations around the destination (Figure 8):

$$x^{[k]}_t \sim p(x_t|x^{[k]}_{t-1}, u_t) \qquad (7)$$

2. Landmark Recognition: a new measurement z_t of a nearby landmark m_{c_t} is made at t, and c_t is identified as j $(j \in \{1, ..., N\})$ by the landmark recognition algorithm (to be elaborated in Section VI). If m_j is never seen before, a new landmark is created, with coordinates and orientation computed based on user pose x_t and relative distance, angle in z_t.

3. Landmark Update: If m_j is a known landmark, its states are updated. Assuming the most recent attributes of landmark m_j are μ^{t-1}_j and Σ^{t-1}_j, where $\mu^{t-1}_j = (\mu_x, \mu_y, \mu_\varphi)$ are its coordinates and orientation in the global coordinate system, and Σ^{t-1}_j the corresponding 3×3 covariance matrix.

- *Prediction*. Given a user pose $x_t = (x, y, \varphi)$ at time t and m_j's attributes μ^{t-1}_j at $t-1$, a measurement prediction $\hat{z}_t$ about the relative distance and angle between the user and m_j can be made as:
$$\hat{z}_t = \begin{pmatrix} \hat{d} \\ \hat{\theta} \end{pmatrix} = \begin{pmatrix} \sqrt{(\mu_x - x)^2 + (\mu_y - y)^2} \\ \mu_\varphi - \varphi \end{pmatrix} \qquad (8)$$
simply their differences in coordinates and orientations.
- *Observation*. Given m_j's image, the localization algorithm (Section III) generates multiple hypotheses of (d, θ), each with a weight. We pick one hypothesis at probabilities proportional to their weights as the actual measurement $z_t = (d, \theta)^T$.
- *Extended Kalman Filter* (EKF) [9]. It linearizes the measurement model (Eqn. 8) such that measurement errors become linear functions of noises in user pose and landmark attributes. Then it computes the "optimal" distribution of hidden variables (e.g, landmark attributes) given observations, such that the discrepancies between predicted and actual measurements are minimized.
Step 1: The Kalman gain is computed as:
$$Q = H\Sigma^{t-1}_j H^T + Q_t, K = \Sigma^{t-1}_j H^T Q^{-1} \qquad (9)$$

[2] We also tried to place each trajectory w.r.t. all previous ones but find the much increased complexity brought only marginal improvements. Thus we use the much simpler method as in Eqn 6.

where Q_t is a 2×2 covariance of Gaussian measurement noises in (d, θ), H is the 2×3 Jacobian matrix of $\hat{z}_t$, with elements partial derivatives of $(\hat{d}, \hat{\theta})$ w.r.t. $(\mu_x, \mu_y, \mu_\varphi)$.

Step 2: The mean and covariance of m_j are updated as:

$$\mu_j^t = \mu_j^{t-1} + K(z_t - \hat{z}_t), \Sigma_j^t = (I - KH)\Sigma_j^{t-1} \quad (10)$$

where I is a 3×3 unit matrix.

Figure 8 shows that after the update, the uncertainties (quantified by covariances represented in oval sizes) in a landmark's location and orientation become less and the distributions become more concentrated. To simplify the wall length estimation, we use an weighted average of $(w_L^{t-1}(t-1) + w_L)/t$ as the updated wall length w_L^t for landmark m_j (w_R computed similarly). We find the results are sufficiently accurate.

4. Weight Update: we assign each particle k a weight that quantifies the probability (Eqn. 11) that the actual measurement z_t can happen under the user pose $x_t^{[k]}$ and updated landmark states (μ_j^t, Σ_j^t). The larger the probability, the more likely that the estimated user pose and landmark attributes are accurate.

$$\begin{aligned} w^{[k]} &= p(z_t | x_t^{[k]}, m_j) \\ &= |2\pi Q|^{-\frac{1}{2}} exp\{-\frac{1}{2}(z_t - \hat{z}_t)^T Q^{-1}(z_t - \hat{z}_t)\} \end{aligned} \quad (11)$$

Under Gaussian noises and linearization approximation [20], the weight can be computed in closed form of the actual measurement z_t and its prediction $\hat{z}_t$. A prediction $\hat{z}_t$ closer to actual z_t leads to a larger weight.

5. Resampling: After the weights for all particles are computed, a new set of particles is formed by sampling K particles from the current set, each at probabilities proportional to their weights. The above steps are repeated on the new set for the next time slot.

VI. Landmark Recognition

Landmark recognition detects which landmark is measured in the current data sample: a new one never seen before, or an existing one already known. Incorrect recognition will cause wrong updates, thus possibly large errors or even incorrect map topology. We take advantage of multiple sensing modalities of complementary strengths for robust recognition: images capture the appearances; poses depict the spatial relationships, and WiFi identifies radio signatures.

Image Based Recognition. Given a test image, we extract its features and compare with those from images of existing landmarks, then determine whether it is a new or existing one. We use a standard image feature extraction algorithm [18] to generate robust, scale-invariant feature vectors. Then we identify matched feature vectors to those from an existing landmark's image. The image similarity S_j^{image} to each existing landmark j is computed as the fraction of matching ones among all distinct feature vectors in the test image and landmark j's image.

Wi-Fi Based Recognition. Although image features distinguish complex landmarks well (e.g. stores and posters), they are ineffective in homogeneous environments such as office and lab, where doors have very similar appearances. We use the *cosine distance* (i.e. the cosine value of the angle between two vectors of WiFi signatures) to quantify the radio signature similarity S_j^{wifi} between the test data and landmark j's data.

Pose Based Recognition. Given the user pose x_t and landmark attributes (e.g., coordinates and orientation), a relative distance/orientation $\hat{z}_t$ can be predicted from Equation 8. The correct landmark j should make this prediction very close to the actual measurement. Based on this intuition, we use the conditional probability that z_t can occur given x_t and m_j's location/orientation as the metric S_j^{pose}, which is exactly the same as weight $w^{[k]}$ in Equation 11.

Aggregate Similarity. An aggregate similarity is computed as $S_j^{image} \cdot S_j^{wifi} \cdot S_j^{pose}$. Since images, WiFi and inertial data are independent from each other, the probability the landmark being j is proportional to the product of the three similarity scores. The product form implies that a small score in any of the three is a strong indication of incorrect match, and the true match would have high scores in all the three.

Using the shopping mall as an example, we observe that the recognition using any individual modality can fail: e.g., pose/WiFi for nearby landmarks, and image for glass walls or similar appearances. Aggregating them, however, achieves almost perfect recognition (more results in Section VIII).

VII. Compartment Estimation

Besides landmarks, a complete floor plan includes also accessible compartments such as hallways and rooms. A commonly adopted technique is occupancy grid mapping [27]: divide the floor into small cells and accumulate evidence on each cell's accessibility to identify compartments. While existing work [4], [14] uses plenty of trajectories, we have only a handful, too sparse to infer accessible areas directly. We make two adaptations to compensate data sparsity: 1) instead of a fixed confidence in cells, we spread attenuating confidences away from trajectories and detected walls; 2) we leverage regions between the camera and landmarks to infer large open regions.

Hallway and Room Shapes. Since only a few trajectories are gathered, they are too sparse to cover all accessible areas. We assign each cell a confidence that increases as it gets closer to a nearby trace or wall segment, because cells closer to traces or walls are more likely accessible. Areas traversed by multiple traces will accumulate more confidence, thus more likely to be accessible. We use a closed loop walking inside each room to reconstruct its shape, and leverage landmark recognition to associate such traces with respective rooms and place their contours on the map.

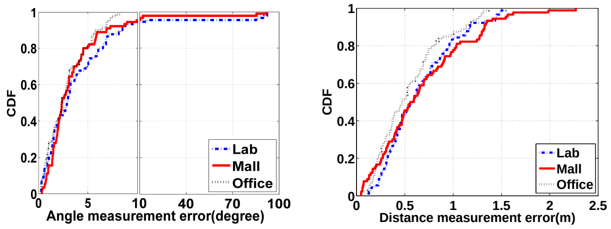
Large Open Regions. Large open regions (e.g., lobbies) need many traces to cover its cells. We leverage the images to infer their sizes. Since the user needs to ensure the landmark is not occluded by obstacles, the region between the camera and the landmark is usually accessible. Thus we compute the triangle region between the camera and landmark (including adjacent wall segments), and assign a fixed confidence to all cells in this area.

VIII. Performance Evaluation

A. Methodology

We use iPhone 5s to collect inertial and image data, and Samsung Galaxy S II for WiFi scans. [3] We define landmarks as store/room entrances, and conduct experiments in three

[3] iOS public API does not give WiFi scan results.



(a) Angle errors (b) Distance errors

Fig. 9. Errors in image measurements.

environments: a 90 × $50m$ office, a 80 × $50m$ lab building and a $140 \times 50m$ shopping mall, with 16, 24, 18 doors/posters as landmarks respectively.

We evaluate Knitter's resilience with three user groups: *dedicated users* who are well trained (i.e., ourselves); 15 *novice users* who spend $5\,\text{min}$ practicing data collection following two simple guidelines: 1) take images from medium distances and angles (e.g., ∼5 meters, ∼ 45°), with the landmark at the center; 2) during walking, hold the phone steady; and 15 *untrained users* who may not follow the guidelines. Feedback from trained ones suggest the two guidelines are easy to follow in practice.

B. Evaluation of Individual Components

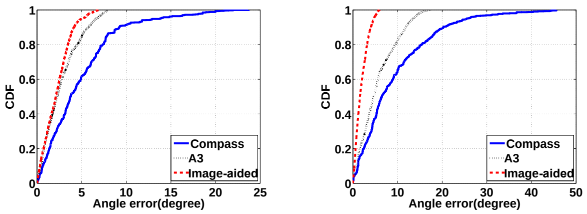
Image Measurements. We first evaluate the accuracy of user locations relative to the landmark, i.e., the extracted distance d and angle θ in Section III. Figure 9(a) and Figure 9(b) show the distribution of angle and distance errors from images in three environments. We observe that the angle measurement errors are around 5°, and that of distance within $1m$, both at 80-percentile. The maximum angle and distance errors are about 94° and 2.2 meters (due to incorrect floor-wall boundary detection). The results show that image extraction in general has high accuracy, but large outliers are possible. Thus we select the top 3 candidates for floor-wall boundary, and compute respective distances/angles, wall segment lengths and weights to form multiple hypotheses as input to map fusion.

Trajectory Angle Calibration. We compare the image-aided calibration method against raw compass or gyroscope readings, and a recent phone attitude A^3 [30] method. We perform experiments in two environments with little/strong magnetic disturbances, both for an 8-minute walking with multiple turns and images.

Figure 10(a) shows the angle error CDF with little magnetic disturbances. We observe that both A^3 and image-aided calibration achieves accurate angle estimations (∼ 5° at 90-percentile, maximum 8°). Raw gyroscope readings (curve omitted due to space limit) suffer linear drifts and reach 32° angle errors after the 8-minute walk, and compass has around 10° at 90-percentile.

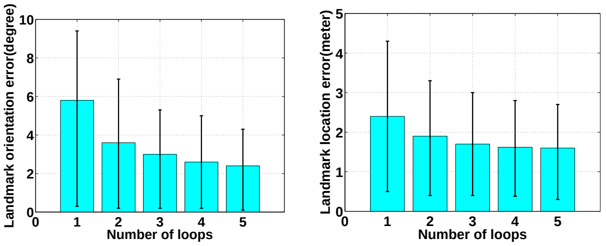
However, when magnetic disturbances are strong (e.g., 90-percentile compass errors around 20° in Figure 10(b)), the errors from A^3 increases (∼ 12° at 90-percentile, maximum 17°) due to frequent and strong disturbances thus incorrect calibrations. The image-aided method remains unaffected and still achieves accurate angle estimation. This demonstrates the robustness of the image-aided calibration method in different environments.

Landmark Recognition. Table I shows the landmark recognition accuracy for 5 loops' data in all three environments. We observe that image-based recognition works well in the mall, but completely fails in office or lab because the landmarks (e.g., doors) appear almost the same. The results after aggregating all valid modalities are 100%, 95.8%, and 100%, proving their complementary strengths.



(a) Without strong magnetic disturbances (b) Under strong magnetic disturbances

Fig. 10. Angle errors without and under strong magnetic disturbances.



(a) Orientation errors (b) Location errors

Fig. 11. Landmark placement errors with different number of loops data.

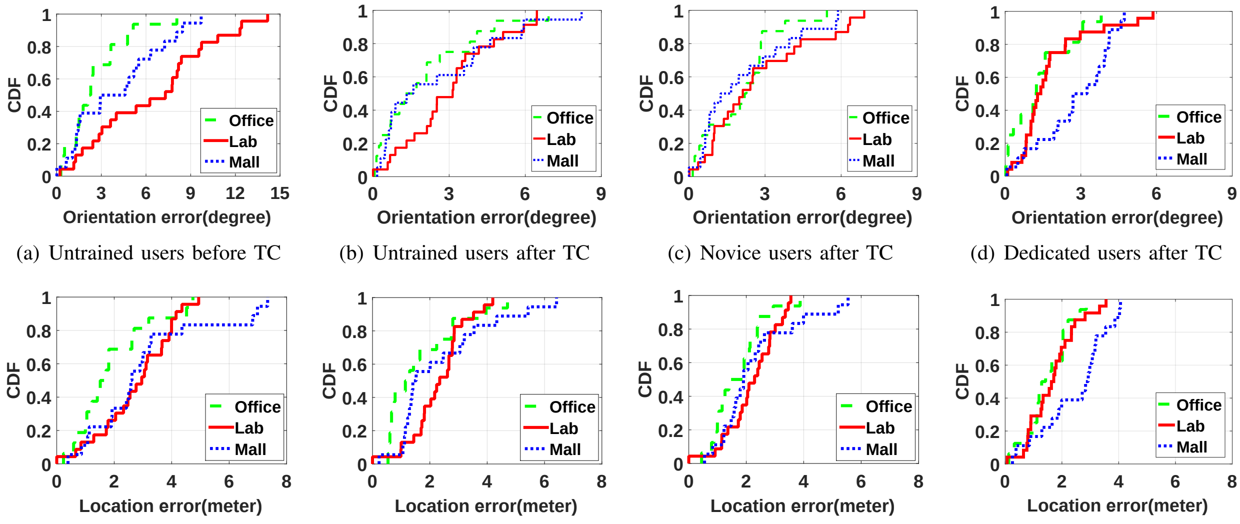
TABLE I
LANDMARK RECOGNITION ACCURACY

	Office building	Lab building	Shopping Mall
Image	–	–	91.7%
WiFi	89.1%	87.5%	79.2%
Pose	100%	86.2%	86.1%
All sensors	100%	95.8%	100%

C. Map Fusion Framework

Landmark Update Performance. Figure 11 shows the changes in maximum, mean and minimum landmark orientation and location errors as more loops' data are used for office (the other two are similar). We observe that more data reduce errors: e.g., the maximum errors drop from 9.4° to 4.3°, and 4.3m to 2.7m. Also 3 loops seem sufficient: the mean errors (3° and $1.7m$) do not further improve much. Thus we do not need many loops in each environment.

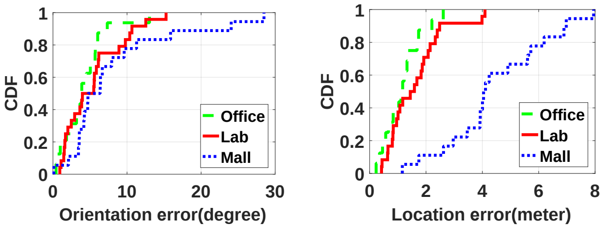
Untrained, Novice and Dedicated Users. The final orientation and location errors of landmarks from untrained users are shown in Figure 12, before (Figure 12(a)(e)) and after (Figure 12(b)(f)) trajectory cleaning (TC). Figure 12(c)(g) show the final results for novice users, and Figure 12(d)(h) show those for dedicated users. We make several observations: 1) untrained users have much larger errors (Figure 12(a)(e)), e.g., 4° ∼ 12° and $5 \sim 7m$ errors at 90-percentile before trajectory cleaning. 2) Trajectory cleaning is quite effective for both untrained and novice users. E.g., it cuts down orientation errors by 6° and location errors by $2m$ for untrained users at 90-percentile (Figure 12(b)(f)). 3) after trajectory cleaning, novice users (Figure 12(c)(g)) achieve accuracies comparable to dedicated users (slightly higher 4° ∼ 6° vs. 3° ∼ 5° and $3 \sim 5m$ vs. $2 \sim 4m$ at 90-percentile), and untrained users have about 2° and $2m$ more in maximum error.



(a) Untrained users before TC (b) Untrained users after TC (c) Novice users after TC (d) Dedicated users after TC

(e) Untrained users before TC (f) Untrained users after TC (g) Novice users after TC (h) Dedicated users after TC

Fig. 12. Final landmark orientation and location errors for untrained, novice and dedicated users. (a)(b)(e)(f) for untrained users before (1st column) and after (2nd column) trajectory cleaning (TC). (c)(g) for novice users, and (d)(h) for dedicated users.



(a) Orientation errors (b) Location errors

Fig. 13. Landmark orientation and location errors using top hypothesis only.

Examination shows that larger errors from untrained users are mainly caused by careless or impatient data collection, e.g., not holding the phone steady, swinging the phone, changing stride lengths suddenly, and taking photos under extreme bright/dark lights or with motion. While novice users exhibit more care and their data have better quality, thus achieving results comparable to dedicated users. This shows the resilience of Knitter: a novice user with a few minutes' training can produce quality maps.

Multi-hypothesis Measurement. Although the image measurement is shown to be quite reliable, incorrect boundary line can cause occasional large errors. Figure 13 show the errors using top hypothesis only. Compared to Figure 12 where all hypotheses are used, the orientation errors increase significantly (e.g., maximum from 6° to 28°), so do location errors (especially for the mall, maximum from 4m to 8m). Due to many visual disturbances (e.g., decoration strips on the floor, glass windows and doors) in complex environment like malls, incorrect boundary lines can become the top hypothesis and cause large outliers. In simpler environments like office, image extraction is more robust. Thus errors do not increase as much when only the top hypothesis is used.

D. Map Overall Shapes

The reconstructed maps from 5 loops' data gathered by novice users and their respective ground truth floor plans are shown in Figure 14. We can see they match the ground truth quite well. To quantify how accurate the shape of a reconstructed map is, we overlay it onto its ground truth to achieve the maximum overlap by rotation and translation. We define precision, recall and F-score to measure the degree of overlap:

$$P = \frac{S_{re} \cap S_{gt}}{S_{re}}, R = \frac{S_{re} \cap S_{gt}}{S_{gt}}, F = \frac{2P \cdot R}{P + R}, \qquad (12)$$

where S_{re} denotes the size of reconstructed map, S_{gt} that of its ground truth, and $S_{re} \cap S_{gt}$ that of the overlapping area.

Table II shows the precision, recall and F-score of the three maps. We observe that Knitter achieves high precisions around $85 \sim 90\%$ for all three buildings, high recalls for lab (around 85%), and high F-scores for office and lab around 86%. Recalls are lower than precision (especially the mall) due to small amounts of trajectories, large open regions and unreachable room spaces when walking. We also evaluate the overall shape of maps using data collected by ourselves, and results are similar with slight increase of $3 \sim 5\%$ in precision, recall and F-score. These prove that novice users' data can construct maps on par to dedicated users, and approximate the shapes of ground truths very well.

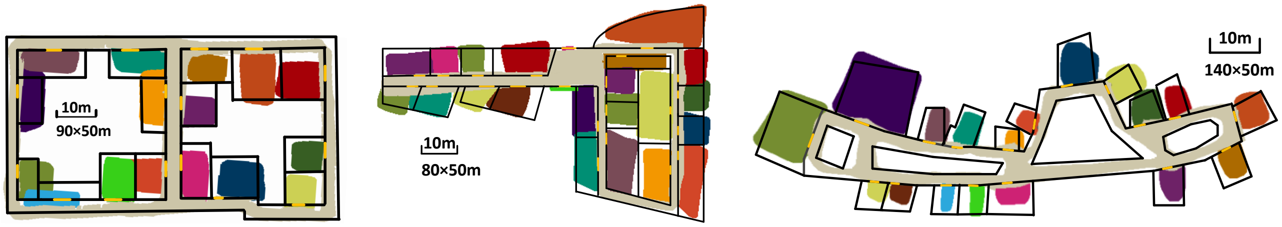
TABLE II
SHAPE EVALUATION OF FLOOR PLANS

	Precision	Recall	F-score
Office building	89.29%	82.62%	85.83%
Lab building	87.73%	85.51%	86.61%
Shopping mall	84.21%	74.30%	78.95%

E. Comparison with Jigsaw

We compare the reconstructed map of Knitter to that of Jigsaw [14], a latest work. Knitter explores a lightweight localization method that requires only one image; it combines multiple sensing modalities to recognize landmarks, and uses Bayesian Networks to incrementally update the map upon each data sample.

In contrast, we find several limitations of Jigsaw. 1) Jigsaw uses Structure from Motion [3], a compute-intensive technique that requires over 100 photos per landmark, thus taking long time and intensive human efforts to collect. 2) It assumes landmarks with distinctive appearances to construct the "point



(a) Office building (b) Lab building (c) Shopping mall

Fig. 14. Reconstructed and ground truth floor plans for the office, lab, and mall.

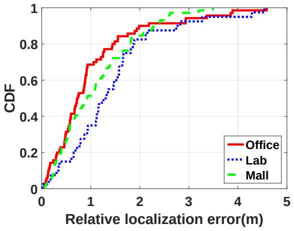


Fig. 15. Relative localization errors using reconstructed maps.

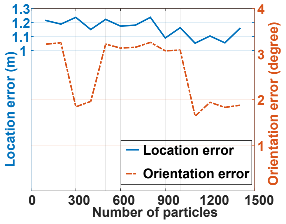


Fig. 16. Landmark errors vs. number of particles.

cloud", which is not applicable in visually homogeneous environments such as office and lab, and it assumes perfect landmark recognition (by image matching [18] or humans). 3) Its maximum likelihood optimization requires many constraints from large amounts of data.

We compare the reconstruction performance of Knitter and Jigsaw for the mall only (because SfM [3] does not work well in office/lab). Since crowdsensing may take long time (weeks or longer) and high expenses to collect large quantities of data, we gather the data by ourselves. It takes us about 21 man-hours to collect the needed data (over $2,400$ images, about 200 hallway and room traces). Then we manually associate images to respective landmarks to ensure perfect landmark recognition. Table III summarizes the comparison results.

TABLE III
COMPARISON WITH JIGSAW

	Jigsaw	Knitter
Effectiveness	Only mall	Office, lab, mall
#Images/landmark	150	$1 \sim 5$
Data collection	21 man-hours	1 man-hour
Orientation accuracy	$4°$	$4°$
Location accuracy	$2m$	$3 \sim 4m$

We observe that Knitter achieves the same orientation accuracy ($4°$ at 80-percentile) as Jigsaw, and slightly higher location errors ($3 \sim 4m$ vs. $2m$ at 80-percentile) which do not constitute too big a challenge for customers because stores are separated much farther away. However, Knitter requires about only 1 man-hour to collect 5 loops' data, only 5% that of Jigsaw's 21 man-hour efforts. The batch optimization in Jigsaw is also susceptible to outliers. We find sometimes a single large outlier can skew landmark locations by over $10m$.

The comparison shows advantages of Knitter: lightweight algorithms speeding up data collection by more than $20\times$; trajectory cleaning ensuring data quality from novice users; a multi-hypothesis, incremental map fusion scheme for accurate map updates and tolerance of residual errors; reliable landmark recognition based on multi-modality sensing.

F. Miscellaneous

Reconstructed Maps for Localization. One major usage for reconstructed floor plans is to pinpoint user locations on maps. We select 80 random test locations in each environment; users stand at each test location and take a photo of the closest landmark. During localization process, first we collect the inertial data, WiFi signatures and images to recognize the landmark, then employ our single image localization algorithm (Section III) to compute the user's relative location to the landmark.

Figure 15 shows CDFs of the relative position errors (distance between the computed and true relative locations to the correct landmark) in all three environments. For practical purposes such as navigation, accurate relative locations to a correct landmark is sufficient to produce proper routes on the map. We observe that the 90-percentile position errors are around $2.0m$, $2.8m$ and $2.3m$ in office, lab and mall, respectively. The large errors in lab are due to landmark recognition mistakes, since its landmarks (e.g., doors) have similar appearances and are close to each other. The mall has almost perfect recognition but larger sizes, thus intermediate errors. Although not yet a full-fledged solution, the above demonstrates the potential of reconstructed maps for localization.

Number of Particles. More particles in general improve the mapping accuracy but increase computing time. Figure 16 shows that the average errors decrease slightly (from $1.2m/3°$ to $1.1m/2°$) and become stable after 1000 particles. [4] The computation time increases from 54s with 100 particles to 292 seconds with 1000 particles for 5 loops update, still very small. This shows even with small number of particles we can achieve accurate results.

Energy. We use Monsoon Power Monitor [2] and find that one-time image-taking plus WiFi-scan cost around 25 Joules. For a typical indoor environment with 20 landmarks, the 20 images and 20 WiFi scans at photo locations cost 500 Joules. Transmitting all data ($\sim 5MB$ for 800×600 images, inertial and WiFi data) costs about 5 Joules on WiFi [6]. Compared to the battery capacity of $21k$ Joules [1], the data sensing and transmission consume about 2.4% of the phone's battery.

IX. RELATED WORK

Indoor Floor Plans. Indoor floor maps is a relatively new problem in the mobile community. CrowdInside [4] uses inertial data to construct user trajectories to approximate shapes of accessible areas. Jigsaw [13], [14] combines vision and mobile techniques to generate accurate floor plans using many images. Walkie-Markie [25] identifies when the WiFi signal strength reverses the trend and uses them as calibration points to construct hallways. Jiang et. al. [16] detect room and

[4] The dip in orientation error around $300 \sim 500$ particles is due to some outliers temporarily filtered out. They are permanently filtered out beyond 900 particles.

hallway adjacency from WiFi signature similarity, and combine user trajectories to construct hallways. MapGenie [21] leverages foot-mounted IMU (Inertail Measurement Unit) for more accurate user trajectories. Shin et. al. [26] use mobile trajectories and WiFi signatures in a Bayesian setting for hallway skeletons. Sankar et. al. [24] combines smartphone inertial/video data and manual user recognition to recover room features and model the indoor scene of Manhattan World (i.e., orthogonal walls). IndoorCrowd2D [7] generates panoramic indoor views of Manhattan hallway structures by stitching images together.

Compared to them, our distinction is fast, accurate, resilient map construction with a single random user. We produce maps with qualities comparable to the latest method [14], and more than 20× speed up. We also propose incremental map construction utilizing multi-hypothesis inputs and robust landmark recognition, which are suitable for sparse data.

Vision-based 3D Reconstruction. Structure from Motion [3] is a famous technique for scene reconstruction. It creates a "point cloud" form of object exterior using large numbers of images from different viewpoints. iMoon [8] and OPS [19] use it for navigation and object positioning.

Indoor floor plan is essentially a 2D modeling problem that requires reasonably accurate sizes, shapes of major landmarks, but not uniform details everywhere, which is the strength of 3D reconstruction. Compared to them, our focus is not on vision. We carefully leverage suitable techniques for a novel localization method using a single image, thus deriving landmark geometry attributes. We leverage much lighter weight mobile techniques to process inertial and WiFi data for reasonably accurate floor maps with much less data and complexity.

SLAM (Simultaneous Localization And Mapping) estimates the poses (usually 2D locations and orientations) of the robot and locations of landmarks (mostly feature points on physical objects) in unknown environments. Some recent work [11], [12], [28] have used sensors in commodity mobile devices but mostly focus on localization, not map construction.

Compared to them, we must extract information and create complete maps reliably despite low quality and quantity data from common users. The precision and variation of sensor data from commodity mobile devices are far worse than those from special hardware in robotics. We also need to filter, fuse fragmented and inconsistent data from random users.

X. Conclusion

We propose Knitter, which constructs accurate indoor floor plans requiring only one hour's data collection by a single random user. Compared to the latest work, Knitter creates maps of similar quality with more than 20× speed up. Its speed and resilience come from novel techniques including single image localization, multi-hypothesis input, trajectory calibration and cleaning methods, and fusion of heterogeneous data's results using an incremental map construction framework that updates map layouts based on measurement evidences. Extensive experiments in three different large indoor environments for 30+ users show that a novice user with a few minutes' training can produce complete and accurate floor plans on par to dedicated users, while incurring only one man-hour's data-gathering efforts.

In the future, we plan to investigate methods to measure the landmarks without distinct or flat facades, and leverage magnetic signatures and WiFi prorogation models to improve the recognition accuracy.

References

[1] iphone 5s spec. https://en.wikipedia.org/wiki/IPhone_5S.

[2] Power Monitor. https://www.msoon.com/LabEquipment/PowerMonitor.

[3] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *Communications of the ACM*, pages 105–112, 2011.

[4] M. Alzantot and M. Youssef. Crowdinside: Automatic construction of indoor floorplans. In *SIGSPATIAL*, pages 99–108, 2012.

[5] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[6] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. In *USENIX ATC*, 2010.

[7] S. Chen, M. Li, K. Ren, X. Fu, and C. Qiao. Rise of the indoor crowd: Reconstruction of building interior view via mobile crowdsourcing. In *ACM SenSys*, 2015.

[8] J. Dong, Y. Xiao, M. Noreikis, Z. Ou, and A. Ylä-Jääski. imoon: Using smartphones for image-based indoor navigation. In *ACM SenSys*, 2015.

[9] G. Einicke and L. White. Robust extended kalman filtering. *IEEE Transactions on Signal Processing*, 1999.

[10] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *AAAI KDD*, pages 226–231, 1996.

[11] R. Faragher and R. Harle. Smartslam - an efficient smartphone indoor positioning system exploiting machine learning and opportunistic sensing. In *ION GNSS+*, 2014.

[12] R. Gao, Y. Tian, F. Ye, G. Luo, K. Bian, Y. Wang, T. Wang, and X. Li. Sextant: Towards ubiquitous indoor localization service by photo-taking of the environment. *IEEE Transactions on Mobile Computing*, 15(2):460–474, 2016.

[13] R. Gao, M. Zhao, T. Ye, F. Ye, G. Luo, Y. Wang, K. Bian, T. Wang, and X. Li. Multi-story indoor floor plan reconstruction via mobile crowdsensing. *IEEE Transactions on Mobile Computing*, 15(6):1427–1442, 2016.

[14] 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 *ACM MobiCom*, pages 249–260, 2014.

[15] D. Gusenbauer, C. Isert, and J. Krosche. Self-contained indoor positioning on off-the-shelf mobile devices. In *IEEE IPIN*, 2010.

[16] Y. Jiang, Y. Xiang, X. Pan, K. Li, Q. Lv, R. P. Dick, L. Shang, and M. Hannigan. Hallway based automatic indoor floorplan construction using room fingerprints. In *ACM UbiComp*, pages 315–324, 2013.

[17] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *IEEE CVPR*, pages 2136–2143, 2009.

[18] D. G. Lowe. Object recognition from local scale-invariant features. In *IEEE ICCV*, 1999.

[19] J. Manweiler, P. Jain, and R. R. Choudhury. Satellites in our pockets: An object positioning system using smartphones. In *ACM MobiSys*, 2012.

[20] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *AAAI*, pages 593–598, 2002.

[21] D. Philipp, P. Baier, C. Dibak, F. Drr, K. Rothermel, S. Becker, M. Peter, and D. Fritsch. Mapgenie: Grammar-enhanced indoor map construction from crowd-sourced data. In *IEEE PerCom*, pages 139–147, 2014.

[22] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: Zero-effort crowdsourcing for indoor localization. In *ACM MobiCom*, pages 293–304, 2012.

[23] C. Rother. A new approach to vanishing point detection in architectural environments. In *BMVC*, pages 382–391, 2000.

[24] A. Sankar and S. Seitz. Capturing indoor scenes with smartphones. In *ACM UIST*, pages 403–412, 2012.

[25] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, and Y. Zhang. Walkie-markie: Indoor pathway mapping made easy. In *USENIX NSDI*, 2013.

[26] H. Shin, Y. Chon, and H. Cha. Unsupervised construction of an indoor floor plan using a smartphone. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 42(6):889–898, 2012.

[27] S. Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous robots*, 15(2):111–127, 2003.

[28] Y. Tian, R. Gao, K. Bian, F. Ye, T. Wang, Y. Wang, and X. Li. Towards ubiquitous indoor localization service leveraging environmental physical features. In *IEEE INFOCOM*, pages 55–63, 2014.

[29] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: Unsupervised indoor localization. In *ACM MobiSys*, pages 197–210, 2012.

[30] P. Zhou, M. Li, and G. Shen. Use it free: Instantly knowing your phone attitude. In *ACM MobiCom*, pages 605–616, 2014.