Enhancing WiFi-based Localization with Visual Clues

Han Xu[‡], Zheng Yang[†], Zimu Zhou[‡],Longfei Shangguan[‡], Ke Yi[‡], and Yunhao Liu[†]
[‡]CSE Department, Hong Kong University of Science and Technology
[†]School of Software and TNList, Tsinghua University
{hxuaf, zzhouad, lshangguan, yike}@cse.ust.hk, {yang, yunhao}@greenorbs.com

ABSTRACT

Indoor localization is of great importance to a wide range of applications in the era of mobile computing. Current mainstream solutions rely on Received Signal Strength (RSS) of wireless signals as fingerprints to distinguish and infer locations. However, those methods suffer from fingerprint ambiguity that roots in multipath fading and temporal dynamics of wireless signals. Though pioneer efforts have resorted to motion-assisted or peer-assisted localization, they neither work in real time nor work without the help of peer users, which introduces extra costs and constraints, and thus degrades their practicality. To get over these limitations, we propose Argus, an image-assisted localization system for mobile devices. The basic idea of Argus is to extract geometric constraints from crowdsourced photos, and to reduce fingerprint ambiguity by mapping the constraints jointly against the fingerprint space. We devise techniques for photo selection, geometric constraint extraction, joint location estimation, and build a prototype that runs on commodity phones. Extensive experiments show that Argus triples the localization accuracy of classic RSS-based method, in time no longer than normal WiFi scanning, with negligible energy consumption.

Author Keywords

Indoor Localization; Smart Phone; Photogrammetry

ACM Classification Keywords

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

INTRODUCTION

The popularity of mobile and pervasive computing has stimulated extensive interests in indoor localization, which is a critical enabler for location-based applications in living, production, commence, and public services. Among various indoor localization methodologies, Received Signal Strength (RSS)-based localization systems have attracted much attention in recent years and have been extensively studied as the most promising and relatively inexpensive solution for indoor positioning [25]. Their methods basically depend on detecting and analyzing the signals of the widely deployed WiFi infrastructures in public indoor environment and the integrated WLAN card in most off-the-shelf mobile devices. Therefore, these systems require no deploying investment, no additional hardware, and reasonable training overhead, which make it very appealing for commercialization over other measurement-based methods (e.g., ultrasound [20] and visible light [12]) and pure image-based systems [7].

RSS-based localization scheme typically contains two stages: training and operating. In the training stage, a fingerprint database is constructed by locationally labelled fingerprints collected from on-site survey and calibration. Next in the operating stage, a user sends a location query with his/her current RSS fingerprint, by which localization algorithms query the fingerprint database and return the closest matched fingerprint as well as its corresponding location. However, despite more than one decade of research [4, 5, 13, 21, 28, 33, 36], RSS-based indoor localization has not yet been widespread so far. Google Indoor Map, the industrial state-of-the-art, covers about 10,000 locations [1], which are only a tiny fraction of millions of shopping malls, museums, and airports worldwide. One major obstacle behind the sporadic availability is that the mainstream indoor localization technologies overly rely on wireless signal features, however, such features suffer from dramatic performance degradation in complex situations on account of multipath fading and temporal dynamics [35]. Under this context, this problem is also known as "Fingerprint Ambiguity" [26], where two distinct locations may possess similar RSS fingerprints. Therefore, a user may be mistakenly matched to position distant from the true position. In particular, it was reported that fingerprint ambiguity is the root cause of large errors in WiFi-based localization [14].

In existing literature, many pioneers have been addressing the fingerprint ambiguity and the major efforts have been made in the following two aspects. One aspect is called *Mobility Increases Localizability* [34], where the ambiguity is reduced by utilizing user movement. One popular way is to use particle filter together with sensor fusion [8, 11], in which the ambiguity can be modelled by the particle behavior to some extent. Another way is to directly eliminate candidate fingerprints by tracking user movements [26]. And the other aspect is called *Localization with Peer Assist*, where the ambiguity is reduced by the help from other users. One representative solution [14] is to obtain accurate acoustic ranging estimated among peer phones, then maps their locations jointly against WiFi signature map subjecting ranging constraints. Another type of solution somehow utilizes the opportunistic encounters between peers [21, 23]. However, these aforementioned works, more or less, have the following limitations that impede their real-life deployment: 1) Require user movement trace, resulting in that the user cannot obtain his/her location immediately after launching the app [8, 11, 26]. In addition,

UbiComp'15, September 7–11, 2015, Osaka, Japan.

http://dx.doi.org/10.1145/2750858.2807516

accurate moving trajectory is difficult to be extracted from smart phone built-in sensors. 2) Impose some strong assumptions on user behaviors such as fixed phone orientation [30] and being stationary during acoustic ranging [14]. 3) Require a sufficient number of willing peers [14, 21, 23].

The above limitations motivate us to design and implement Argus [1], an image-assisted positioning scheme. The intuition is simple: recent smart phones are all equipped with powerful cameras and people take photos with their phones anywhere and anytime. It is our vision to reduce the fingerprint ambiguity by extracting geometric constraints by utilizing visual clues beneath those ubiquitous photos. Then by mapping the constraints onto the fingerprint space, we are likely to distinguish multiple locations with similar fingerprints. On this basis, Argus works as follows. When an user wants to know the location, he/she bootstraps Argus app and takes a photo of nearby place of interest (denoted by POI, it can be any physical feature or indoor landmark like shop logos, statues, and billboards). The absolute POI location is not required and users can take photos at a relatively long distance. The query photo together with his/her current RSS fingerprint will be sent to the server. After matching the query photo to the corresponding POI and handling device heterogeneity, we adopt a photo selection mechanism to choose some supporting photos to assist localization. Then we use a Computer Vision (CV) technique called Structure from Motion (SfM [10]) to generate geometric constraints among those photos. Finally, Argus maps their locations jointly against fingerprint space subjecting the geometric constraints and our cost function.

Despite the simple idea, three major challenges underlie the design of Argus: 1) How to effectively construct the image database? Manual configuration requires a great amount of time and effort, and it is a major obstacle for pure image-based solutions [7]. Instead, we adopt crowdsourcing and extract the geometric constraints without knowing their ground truth locations. 2) How to utilize and evaluate geometric constraints for accurate localization? The key here is that the constraints obtained from photos are relative, unknown scaling, translation, and rotation exist from the corresponding locations in the real world [17]. So we transform the problem into a combinatorial optimization problem, and solve it efficiently by pruning and interpolation. 3) As a mobile application level service, Argus should execute in a real-time and energy-efficient manner. We address these by introducing a module driven structure and a set of efficient algorithms. The key contributions of Argus are summarized as follows.

- We identify the opportunity of leveraging crowdsourced unlabelled photos to resolve RSS fingerprint ambiguity. We observe that the relative positions obtained from SfM can act as the geometric constraints for accurate indoor localization. To our best knowledge, this is the first work that provides image-assisted indoor localization in fingerprint space with crowdsourced photos and commodity phones.

- We propose Argus, an easy-to-use and highly accurate indoor localization scheme. It combines geometric constraints with RSS fingerprints to address fingerprint ambiguity. We adopt a crowdsourcing approach to construct an unlabelled image database and design a combinatorial optimization to evaluate location candidates. Such scheme provides Argus with the following desired properties: 1) One click localization, users can be localized with a single photo, which is natural and easy-to-follow. 2) Highly compatible with existing RSS-based localization systems, only extra effort is to construct an image database, which is crowdsourced and automatic. 3) Fast and accurate, it takes advantage of both the RSS-based method and image-based method, improves localization accuracy while bypassing the huge overhead of pure image-based method.

- We fully implemented Argus on Android platforms and conducted extensive experiments in two types of large complex indoor environments: shopping mall and food plaza. Argus triples the localization accuracy achieved by the mainstream RSS-based method, and limits the mean localization error to less than $1m$ while achieves less than $8s$ localization latency.

[1] Argus is a giant with 100 eyes in Greek mythology.

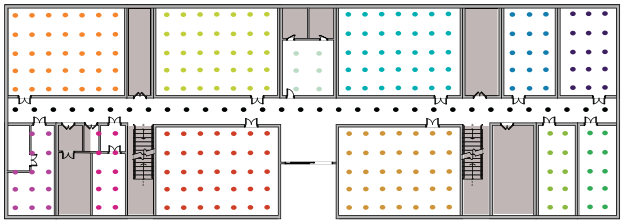


Figure 1. An academic building floor plan, each dot represents a location where the WiFi fingerprint is measured.

The rest of the paper clarifies each of the above contributions, beginning with observation of using WiFi fingerprint for indoor localization, followed by overview, design, implementation, and evaluation of Argus. Finally, we summarize the limitations and point out potential future work.

OBSERVATION OF USING WIFI FINGERPRINT FOR INDOOR LOCALIZATION

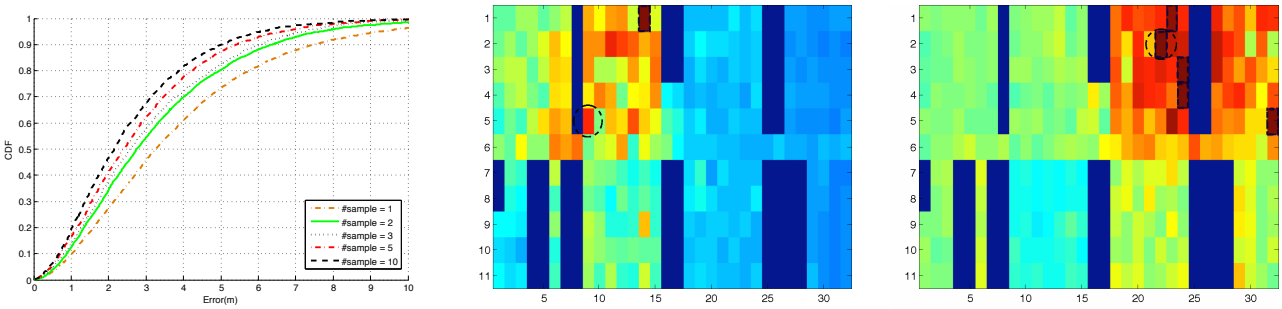
Methodology and Experimental Setup

We conduct this observation in a similar way as classic RSS-based localization system [5]. The major difference is the way we weight average the locations of a few nearest fingerprints in signal space [**?**]. Formally, for each location query fingerprint f_i, the N fingerprints in the database having the smallest distances to it form its neighbor set $\mathbb{N}(i)$. Let $\left[f_i^{\mathbb{N}(1)}, ..., f_i^{\mathbb{N}(N)}\right]$ be its set of N nearest neighbors in signal space. The neighborhood weights of f_i are calculated by:

$$\textbf{Minimize} \quad \left| f_i - \sum_{j \in \mathbb{N}(i)} W_{ij} f_i^{\mathbb{N}(j)} \right|$$

$$\textbf{Subject to} \quad \sum_{j \in \mathbb{N}(i)} W_{ij} = 1$$

We conduct the observation in an academic building as in Figure 1. The building is of size $70 \times 23m$, which contains 16 offices, of which 5 are large rooms of $142m^2$, 7 are small



(a) Errors with different sample size (b) Large Error Case 1 (c) Large Error Case 2

Figure 2. Observation Analysis

ones with different sizes and the other 4 are inaccessible. The phone takes 50 WiFi fingerprints at each of the 279 known locations from 26 APs. Each location can receive signals from $6-7$ APs on average. 20 samples in each location are utilized for training stage, and the remaining 30 samples are left for operating stage. We consider the top 5 nearest neighbors in this experiment throughout the paper.

Result Analysis

First of all, we investigate the WiFi localization performance under different sample sizes, which are the numbers of WiFi fingerprints the phone collects before sending a query. The localization results are presented as a cumulative distribution function of localization errors as illustrated in Figure 2a. Evidently, more samples contribute to more reliable localization results, but at the cost of higher energy consumption and latency. Besides that, even if 10 samples are in use, large errors occupy a considerable part (see long tails of CDF curves exhibiting errors more than $5m$). Large errors are troublesome in many cases, such as leading the users into a wrong shop or instructing them to turn at a wrong corner.

Given the above observations, we try to figure out the root cause of large errors by examining those queries that lead to large errors. Without loss of generality, we roughly divide them into two cases. First, locations far away from the true user position possess more similar fingerprints in signal space. An example of this scenario is illustrated in Figure 2b. The floor plan is divided into grids corresponding to Figure 1. The color of each grid represents its RSS Euclidean distance to the query WiFi fingerprint in signal space (The redder, the closer). A query is sent from location $(9, 5)$ (marked by a dotted circle), its closest neighbor in signal space is $(14, 1)$ (marked by a dotted rectangle). It mainly contribute to a large error of $8.6m$. Second, even if the closest fingerprint is correctly matched, the top closest neighbors in signal space may scatter in physical space. An instance can be seen in Figure 2c, where the query fingerprint is taken at $(22, 2)$, and its top 5 nearest neighbors in signal space are $(22, 2), (32, 5), (23, 1), (24, 4)$, and $(24, 3)$. With the estimated location as the weighted average of the top 5 nearest neighbors, this instance still has a large error of $6.46m$.

In summary, large errors are mainly caused by WiFi fingerprints that are close in signal space but distant in physical space, or in other words, fingerprint ambiguity. However, the reasons for such ambiguity are multi-folds, both the permanent environmental settings (e.g., walls and furniture) and the temporal dynamics (e.g., pedestrians and wireless interferences) can affect the radio propagation and reception [14], thus produce similar fingerprints. These factors always exist in reality and are impossible to eliminate by pure signal processing technique. In this paper, we propose Argus and try to solve the ambiguity from a different perspective by utilizing geometric constraints induced from ubiquitous photos.

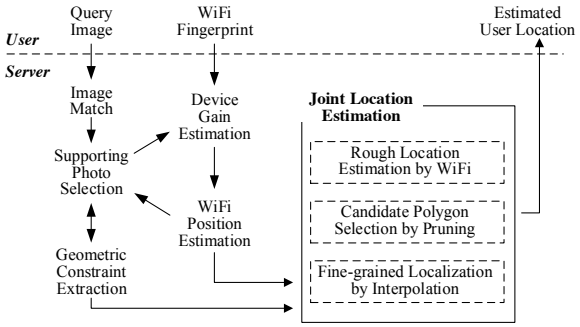


Figure 3. System Overview

SYSTEM OVERVIEW

Working Flow from User's Perspective

When a user needs to be localized, he/she just activates Argus' app on his/her smart phone. The camera is automatically turned on, along with the WiFi scanning. Then the user is instructed to shoot a POI near the center of his/her viewfinder without walking near it (The user can take a photo at a relatively long distance as long as the POI is clearly taken). Afterwards, Argus will send the photo as well as the RSS fingerprint to the server. After the remote processing, a location tag will be returned to the smart phone, and Argus displays this location to the user.

Working Flow from Server's Perspective

As shown in Figure 3, the input of server side are a query image and its corresponding WiFi fingerprint. The server handles the localization query in a pipeline manner briefed as follows. First of all, the server will figure out which POI the user is shooting for, and the photos for the same POI will be identified by a **Image Match** module. Then with a **Supporting Photo Selection** module, it adopts a photo selection mechanism to pick up the most suitable photos (termed as supporting photo) among the photos for the same POI to assist

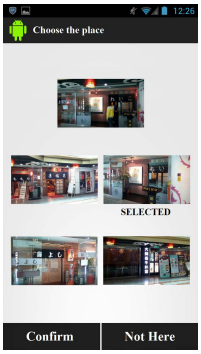


Figure 4. Image Match Correction

human localization. The mechanism considers both the quality of supporting photos and their WiFi fingerprints. Once the supporting photos are selected, a **Geometric Constraint Extraction** module applies a CV technique called Structure from Motion [10] in order to obtain the relative locations and orientations of the shooting sites among these photos (supporting photos + query photo), and these form the geometric constraints for a **Joint Location Estimation** module. At the same time, a rough user localization will be estimated by a **WiFi Position Estimation** module in the same way as we have introduced in the observation after handling the hardware heterogeneity in a **Device Gain Estimation** module. Finally, a **Joint Location Estimation** module estimates user location by considering the WiFi fingerprints and the geometric constraints jointly. We model this joint localization into a combinatorial optimization problem, the position that minimizes the cost function is regarded as the user location.

IMAGE MATCHING

As illustrated in Figure 3, the first thing we do is to identify which POI the user is taking photo for. We implement a direct 2D-to-3D matching framework motivated by the work of [22] using visual words technique for fast image matching. Then, we use a simple threshold-based mechanism to judge whether the POI is correctly identified. If the image matching score is above the threshold, Argus executes the next step directly. However, the photos taken by the users could be unclear, unfocused, blurred, or obstructed. So we adopt an image match correction mechanism when the image matching score is below the threshold. As illustrated in Figure 4, Argus retrieves the top 4 candidate POIs and ask the user to identify the correct one by tapping the thumbnail images. Though the top candidate may not accurately identify the POI, its correctness is largely improved when considering the top 4 candidates. The accuracy improves from 91.3% to 97.7% in the mall and from 87.2% to 95.8% in the plaza, see Table 1. The detail of the dataset will be introduced in the Experiment Section.

OBTAINING RELATIVE POSITION CONSTRAINTS

According to the previous observation, WiFi signal alone is unreliable for accurate indoor localization. In this section, we seek the potential of obtaining geometric constraints from images using a CV technique called SfM. Then we utilize a robust metric denoted by Shape Similarity to measure the performance of using such constraints onto localization. Finally,

Table 1. Image Match Accuracy

#Candidate POIs	Shopping Mall	Food Plaza
Top 1	91.3%	87.2%
Top 2	95.2%	92.3%
Top 3	97.4%	94.6%
Top 4	97.7%	95.8%
Top 5	97.7%	96.1%

we propose a crowdsourced image collection mechanism to construct the image database.

Structure from Motion

SfM [10] is a mature and classic CV technology and it is able to derive a 3D model of objects in the visible scene. The input of SfM is multiple photos of an object from different locations. For each of the photo, SfM runs a feature detection algorithm (e.g., SIFT [15]) and identifies various keypoints. By matching multiple sets of keypoints, SfM attempts to reconstruct: 1) A sparse 3D point cloud of the geometry captured by those keypoints; 2) The relative position and orientation of the camera when the original photos were taken [17]. SfM has been utilized in many localization systems [7, 17, 22], however, users are basically localized by referencing the 3D model. Such mechanism is not suitable in our scenario due to two reasons: 1) Hundreds of overlapping images are required for SfM to compute an accurate dense point cloud for a POI [7, 24]. Without doubt, such volume of images impedes SfM's widespread for indoor localization. 2) Even if the point cloud is completely precise, unknown relative scaling, translation, roll, pitch, and tilt exist from the corresponding locations in the real world [17]. Therefore, image-based localization systems usually require a database recording the relationship with the images and their ground truth locations.

Different from the aforementioned work, we utilize the relative positions among images and localize users by referencing the WiFi fingerprint space. Thus we don't need much overlapping images and we are free of human calibration. More specifically, the input of SfM in our work are the query photo and selected supporting photos. The output relative positions form a polygon, of which the vertices represent the locations where the original photos were taken.

Shape Similarity

The shape similarity metric plays two important roles in our work. One is to measure how good the geometric constraints generated by SfM are and the other one is to help photo selection. As mentioned above, the SfM generates a polygon using the query photo and selected supporting photos. Meanwhile, the ground truth locations of the photos form another polygon with absolute physical distances. The more similar the two polygons are, the more likely the relative position constraints satisfy a kind of geometric constraint for accurate localization. To evaluate its feasibility, we conducted an evaluation by taking photos of POIs in a shopping mall and a railway station. In each scenario, we chose 7 POIs and the number of photos taken for a certain POI ranges from 40 to 120, and more than 1000 photos were taken. We manually recorded the ground truth locations. In this evaluation, we use half photos as training set and the other half as testing set.

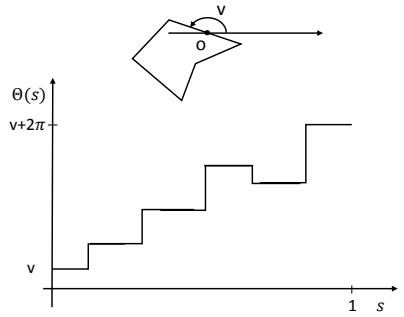


Figure 6. Turning Function

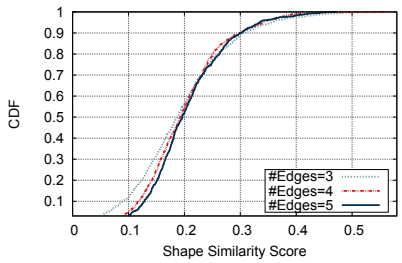


Figure 7. Evaluation of Shape Similarity

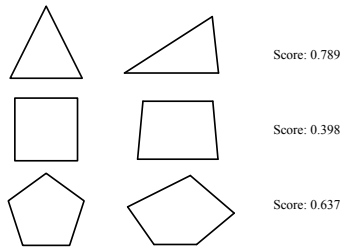


Figure 8. Several Illustrations

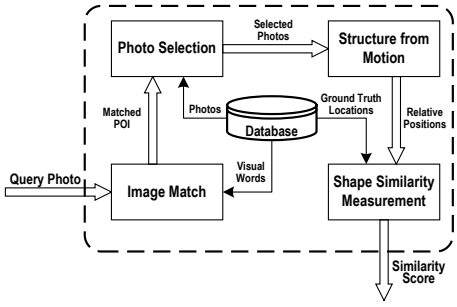


Figure 5. Procedure of Measuring Shape Similarity

Then, each photo in the testing set follows the steps in Figure 5. Firstly, we use image matching technique to identify the POI by retrieving the database. Secondly, among the training photos for the same POI, we randomly choose N photos ($2 \sim 4$ in our case). Thirdly, query photo and the selected N photos will be the input of SfM module, and their relative positions are calculated correspondingly. Fourthly, the Shape Similarity module will measure how similar is the polygon formed by SfM-generated relative positions and that formed by their ground truth locations. Finally, a score indicates the shape similarity will be produced. To be useful in measuring the shape similarity, the score should satisfy the following properties: 1) numeric; 2) invariant to scaling, rotation, and translation, since SfM only produces relative coordinates; 3) computational efficient. Here we mainly modify the method in [3]. There are two key steps in calculating the score:

Representation of Polygons

Instead of representing the polygon using a circular list of vertices, [3] represents the polygon by giving the turning function $\Theta_A(s)$. Here A is the simple polygon and s is the arc length, $\Theta_A(s)$ measures the angle of the counter-clockwise tangent as a function of the arc length s, from some reference point O on A's boundary. Thus $A(0)$ is the angle v that the tangent at the reference point O makes with some reference orientation associated with the polygon (x-axis in our case). $\Theta_A(s)$ keeps track of the turning, increasing with left-hand turns and decreasing with right-hand turns as in Figure 6. Without loss of generality, we rescale each polygon with normalized perimeter length. Note that $\Theta_A(s)$ is invariant to scaling and translation by definition. The rotation of A corresponds to a simple shift of $\Theta_A(s)$ in the θ direction.

The Polygon Distance Function

Given two polygons A and B, and their corresponding turning functions $\Theta_A(s)$ and $\Theta_B(s)$. The shape similarity between A and B can be measured by the distance between the turning function $\Theta_A(s)$ and $\Theta_B(s)$ using L_p distance:

$$\delta_p(A,B) = \|\Theta_A - \Theta_B\|_p = \left(\int_0^1 |\Theta_A(s) - \Theta_B(s)|^p ds\right)^{\left(\frac{1}{p}\right)} \quad (1)$$

However, Equation (1) is sensitive to the rotation of A (or B) and choice of reference point on the boundary of A (or B). Here we make B stable and find the minimum distance by adjusting A. If we rotate A by an angle θ, then the new turning function is given by $\Theta_A(s) + \theta$. And if we move the reference point along the boundary by a distant t, the new function is given by $\Theta_A(s + t)$. Thus we are now going to find the minimum distance by adjusting θ and t, and the shape similarity between A and B is measured by:

$$d_p(A,B) = \left(\min_{\theta\in\Re, t\in[0,1]} \int_0^1 |\Theta_A(s+t) - \Theta_B(s) + \theta|^p ds\right)^{\left(\frac{1}{p}\right)} \quad (2)$$

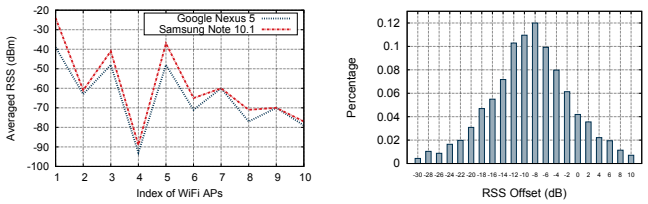
For simple polygons, Equation (2) can be further simplified to a one-variable minimization problem which runs in polynomial time [3]. Thus the similarity score can be calculated efficiently, and people would rate two polygons as resembling each other when their similarity score is less than 0.5 [3].

Evaluation Result

As we can see from Figure 7, most shape similarity score is under 0.5, so the relative positions produced by SfM is qualified for constructing the geometric shape. Furthermore, no matter how many edges (3, 4, or 5) form the polygon, the similarity score distribution is alike, so we may design geometric constraints with few edges. To give an intuitive sense, we list several pairs of polygons in Figure 8 as well as their corresponding similarity scores measured by our algorithm.

Crowdsourced Image Collection Mechanism

Given the above introduction, we use at least 3 photos of the same POI (including the query photo) to construct the geometric constraints. But we hope that the user can be localized by taking a single photo, which means that Argus relies on an established image database. However, the image collection procedure in Argus is simple, since we only need several photos (as few as 3) of the same object from different positions and orientations without knowing their absolute locations. The photos and the WiFi fingerprints when taking the



(a) Collected RSS vectors (b) Measured RSS Offsets

Figure 9. Handling Device Diversity

photos can be uploaded directly to the server. In this paper, we assume the image collection can be crowdsourced to staffs in shopping malls, museums, and airports. Besides that, anyone can volunteer and some incentive mechanisms [32] can be also applied to attract customer to contribute. Last but not least, the first user taking a query photo of a non-exist POI in the database can simply take 2 additional photos and localize himself/herself by the triangle constraint. Then the following users can query the same POI by a single photo.

HANDLING DEVICE DIVERSITY

Before localizing the users with geometric constraints, another practical issue remains: device heterogeneity. Due to the hardware difference (e.g., antenna gain), the same signal may result in different RSS using different devices [16]. Figure 9a illustrates the RSS vectors received by two devices (Google Nexus 5 and Samsung Note 10.1) at the same place in a mall. Figure 9b shows the histogram of RSS offsets measured by the two collocated phones at various locations in the same mall. Such a large device bias can lead to bad localization results if the RSS fingerprint database is constructed with one device yet used by another [13, 16, 19]. one solution is to calibrate the gain offset between any pair of devices offline, which is not scalable [19]. Another solution utilizes the deduction [6, 16] or ratio [9] between AP signals instead of the absolute RSSs, but suffers from large noise and fluctuation in signal power level. In Argus, we employ a learning-based approach to jointly estimate the location and device power level via Expectation Maximization. The idea is similar to [13], yet we adopt different system architecture and metrics.

The detail of our method is as follows. The RSS Euclidean distance between two fingerprints collected at location x ($f(x) = \left[RSS_x^1, RSS_x^2, ..., RSS_x^k\right]$) and y ($f(y) = \left[RSS_y^1, RSS_y^2, ..., RSS_y^k\right]$) in signal space is defined as

$$\triangle_{xy} = \sqrt{\sum_{i=1}^{k} |RSS_x^i - RSS_y^i|^2} \tag{3}$$

Where k is the number of APs. Then in the offset estimation, we simplify the transformation between a pair of devices between A and B as $RSS_B = \alpha RSS_A + \delta$. Furthermore, it was reported that α is close to 1 [19]. So we add a constant δ when calculating the signal space distance, $\triangle_{xy} = \sqrt{\sum_{i=1}^{k} |RSS_x^i - RSS_y^i + \delta|^2}$. By adjusting the value of δ, we can find the minimum signal space distance,

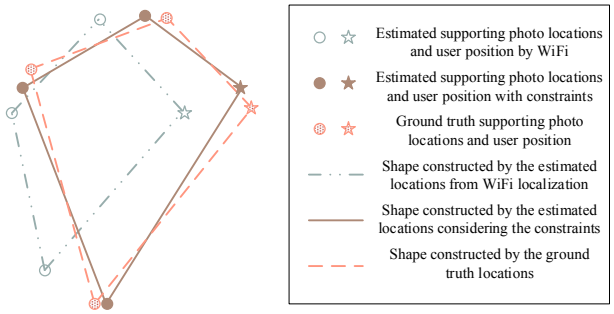


Figure 10. Using geometric constraints to assist localization

the corresponding δ is regarded as the offset. Formally,

$$O_{xy} = arg\min_{\delta} \triangle_{xy} \tag{4}$$

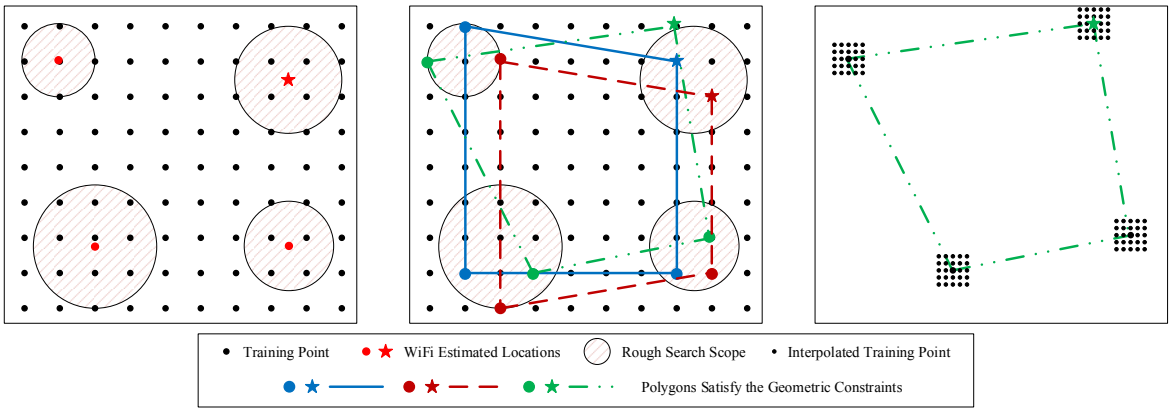
$$\triangle_{xy} = \sqrt{\sum_{i=1}^{k} |RSS_x^i - RSS_y^i + O_{xy}|^2} \tag{5}$$

The above mechanism suffices for online calibration, we evaluate it with data collected by three different devices. Its performance will be presented and discussed in the experiment.

INDOOR LOCALIZATION WITH RELATIVE POSITION CONSTRAINTS

As introduced above, the geometric constraints generated by overlapping photos of the same object are highly accurate. However, the geometric constraints are a relative shape. It needs to be scaled, rotated, and translated to its ground truth location. Unfortunately, there is no absolute location information embedded in the photos, since we don't know where the object is and how far we are away from it. The intuition underlying our algorithm is to construct a shape based on the geometric constraints among supporting photos, and then superimpose the shape onto the fingerprint space on the basis of initial WiFi location estimations. Measured by our cost function, the algorithm scales, rotates, and translates the shape in the fingerprint space, such that the vertices are placed closer to their true locations. The locations where the vertices are placed are regarded as new location estimations. An illustration can be seen in Figure 10. To begin with, each device has a WiFi estimated location (vertices in the dashed-dotted-line graph). Because the geometric constraints are highly accurate to identify the physical shape formed by devices, the solid-line polygon is rather close to that of the ground truth (the dashed-line graph). The additional geometric constraints drive the new user location estimation closer to his/her true location, thus reducing large errors and improving accuracy.

Concretely, the cost function is the sum of RSS Euclidean distances between each device's query WiFi fingerprint and its newly estimated location's WiFi fingerprint (By interpolating nearby training points). To some extent, the function evaluates the collective closeness of all devices jointly. Formally, we denote the shape by $G(E, V)$. V is the vertices of the shape, and their coordinates are generated by SfM. E is the edges of the shape, and $E_{i,j} = (V_i, V_j)$. Let



(a) Rough Location Estimation by WiFi (b) Candidate Polygon Selection by Pruning (c) Fine-grained Localization by Interpolation

Figure 11. Illustration of the heuristic two-tier shape matching

$\{a_1, a_2, ..., a_{|V|}\}$ denote the WiFi estimated locations of vertices in V, and $\{b_1, b_2, ..., b_{|V|}\}$ denote the new locations estimated by our algorithm. Then localization is transformed into the following optimization problem.

$$\textbf{Minimize} \quad \sum_{\forall i} \left[f\left(a_i\right) - f\left(b_i\right)\right]\left[f\left(a_i\right) - f\left(b_i\right)\right]^T$$

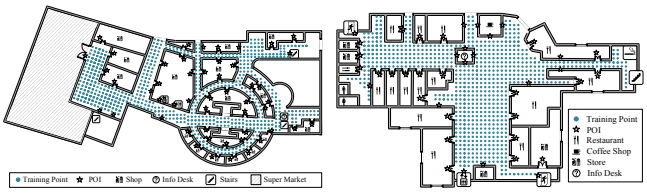
Subject to

$$\forall i, j \quad : \quad (1-\alpha)\frac{|E_{i,j}|}{|E_{1,2}|} \leqslant \frac{|(b_i, b_j)|}{|(b_1, b_2)|} \leqslant (1+\alpha)\frac{|E_{i,j}|}{|E_{1,2}|}$$

where $f\left(x\right)$ is the WiFi fingerprint at location x, $|(b_i, b_j)|$ is the edge length formed by newly location estimates b_i and b_j. α is used to adjust the weight of geometric constraints. We solve this optimization by a heuristic two-tier shape matching algorithm. The key steps are summarized as follows:

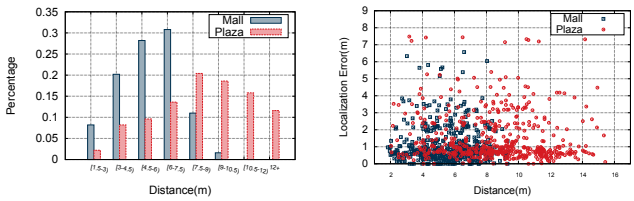
Step 1: Rough Location Estimation by WiFi. Though WiFi fingerprint alone is not accurate enough, it can provide a rough location estimation. Under most circumstances, the true device location is not too far away (say $10m$) from the location a_i estimated by WiFi (e.g., Figure 2a). In Argus, we assume that the true device location is within a small circle C_i centered at a_i with radius r_i during the searching process in the following steps (illustrated in Figure 11a). We empirically set r_i the physical distance from a_i to the corresponding fingerprint's fifth nearest neighbor.

Step 2: Candidate Polygon Selection by Pruning. After restricting the search scope of the possible locations for each device, we try every possible combination of training points to form a valid polygon using a pruning strategy (Figure 11b). The criteria for a valid polygon are the same as the constraints of the above optimization, and $\alpha = 0.2$ in this stage. The rationale of the pruning strategy is that, for two combinations of training points $B^* = \{b_1^*, b_2^*, ..., b_{|V|}^*\}$ and $B' = \{b_1', b_2', ..., b_{|V|}'\}$. If a subset of B^* already costs more than B', then the cost of B^* must be higher than B'. Thus we can sort the training points inside each circle according to its RSS Euclidean distance to the circle center. We try the combination of training points from different circles ascendantly.



(a) Shopping Mall (b) Food Plaza

Figure 12. Floor Plans for Two Experimental Scenarios



(a) Photo Distance Distribution (b) Distance vs. Localization Error

Figure 13. Photo Distance Distribution and Impact on Localization Error

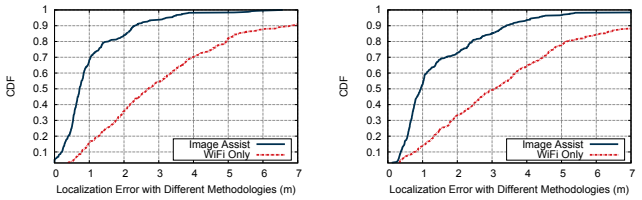
Hence we avoid a large amount of impossible polygons and the top 10 polygons are left for fine-grained localization.

Step 3: Fine-grained Localization by Interpolation. In practice, the consecutive training points often have a constant interval (e.g., $2m$ in our experiment). It is coarse-grained if we limit the user's position to the training points. So we interpolate the fingerprint space around the top 10 polygons as in Figure 11c. The degree of interpolation depends on specific precision. Then, the newly interpolated points are regarded as the training points and we repeat Step 2 with $\alpha = 0.1$ to find the polygon minimizing the objective function.

IMPLEMENTATION AND EXPERIMENTS

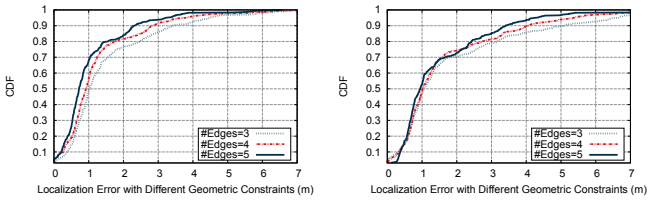
Experimental Setup

The prototype of Argus consists of a front-end and a back-end. We implemented the front-end on three types of mobile devices: Google Nexus 5 phone, Huawei Honor 2 phone, and Samsung Note 10.1 tablet, as a Java extension to the standard Android camera program to ensure the validity and promptness of WiFi fingerprint. During the evaluation, the photos and fingerprints are directly uploaded to our server through



(a) Shopping Mall (b) Food Plaza

Figure 14. Overall Localization Accuracy



(a) Shopping Mall (b) Food Plaza

Figure 15. Impact of the Number of Photos

WiFi connection. Our back-end server is a MacbookPro running Mac OS X 10.9.2, with 2.7GHz Quad-Core Processor and 16GB RAM. We use a mature CV solution SIFT[15] for image feature keypoint extraction. For SfM, we use and modify Bundler [24], an online open source SfM project. And we use VisualSFM [29] to validate and visualize the results.

We conduct localization experiments with the prototype in a shopping mall (Figure 12a) and a food plaza (Figure 12b). These two sites cover diverse human activities (e.g., sitting, eating, and hanging). In each scenario, the WiFi fingerprint database is constructed by on-site survey. We take 10 WiFi fingerprints at each of the training points and the consecutive training points have a distance of $1.8m$. We choose 50 and 41 POIs in the shopping mall and food plaza, respectively, which are marked by pentagrams in Figure 12a and Figure 12b. The number of photos taken for a certain POI ranges from 5 to 20 during the site survey, and more than 1000 photos were taken. In addition, we recruit 2 volunteers to take a total of 200 photos from their own perspectives with two guidelines: One is that the POIs should be outstanding physical features that other people may agree with, and the other is that the POI should be clearly taken near the middle of viewfinder. The ground truth locations when taking photos were recorded manually. The distances between the POIs and the corresponding photos (both from our site survey and volunteers) are illustrated in Figure 13a.

Micro Benchmarks

Overall Localization Accuracy

To measure the overall localization accuracy, we randomly choose half of the photos taken during the site survey to form the image database and the other half are treated as query images (together with the photos from volunteers). We compare our solution (denoted by Image Assist) with classic RSS-based method RADAR [5] (denoted by WiFi Only and 5 samples form a query). As shown in Figure 14, with Argus, the localization accuracy is consistently tripled. Specifically, the 50-percentile error is reduced from $2.73m$ to $0.78m$ in Mall and from $3.01m$ to $0.97m$ in Plaza. The 80-percentile error is reduced from $4.98m$ to $1.38m$ in Mall and from $5.02m$ to $2.30m$ in Plaza. Figure 13b provides a scatter diagram to illustrate the relationship between the distance and the positioning accuracy. The distance between the POI and the photo taken has little impact on the localization accuracy. On the other hand, we give the credit of performance improvement to the joint effect of the following factors (4 supporting photos, with photo selection, and handling device diversity). We evaluate the impacts of these factors subsequently.

Impact of the Number of Photos

Intuitively, the more photos of the same POI we have, the more constraints we obtain. However, more constraints impose more computational complexity to the joint localization algorithm. So we restrict the geometric constraints to polygons with 3, 4, and 5 edges, and the result is shown in Figure 15. Argus benefits from more supporting photos. Yet the gain is negligible when #Edges is more than 5. So we use polygons with 5 edges during the evaluation.

Impact of Photo Selection

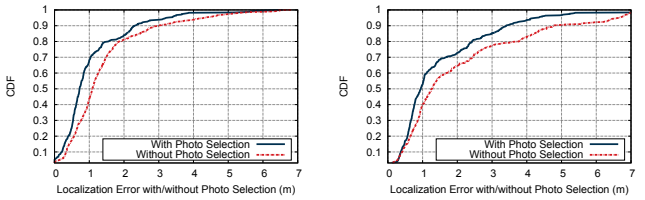
The reasons for photo selection are two-folds: Feasibility, the photos and fingerprints collected by Argus's database accumulate (since the query photos by the previous users can be utilized). Thus there could be hundreds of candidate supporting photos. Since polygons with 5 edges are enough for accurate localization, how to choose 4 supporting photos becomes an issue. Essentiality, the photo quality and WiFi fingerprint quality for the same POI vary sharply (since we cannot guarantee the quality of crowdsourced data). The localization accuracy may fall if we choose "bad" supporting photos. Since the localization relies on both the initial WiFi position estimation and geometric constraints generation, we devise the following 3 criteria for reliable supporting photo selection.

Criterion 1: Compare the WiFi fingerprints of query photo and potential supporting photos. If the ratio of common APs number to all detected APs number is below a threshold, discard the candidate. Then we calculate the standard deviation of common APs after handling the device diversity. If the standard deviation exceeds a threshold, discard the candidate.

Criterion 2: Examine the image similarity between the query photo and candidate supporting photos. If the similarity score is below a threshold, discard the candidate. The rationale is that the supporting photos should have enough overlaps to generate the relative constraints.

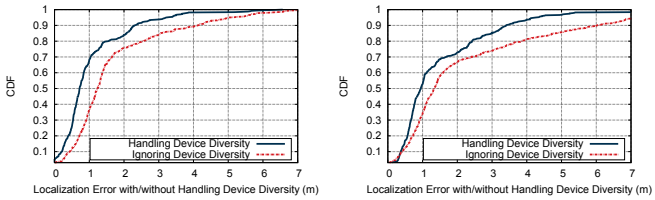
Criterion 3: After filtering some candidates, we examine the remaining combinations (Forming polygons with 5 edges). The key is that, the initial WiFi estimated locations form a polygon and the relative position constraints generated by SfM form another. The more similar two polygons are (using the same metric introduce before), the more likely WiFi fingerprints and geometric constraints work fine together.

The 3 criteria not only consider the quality of WiFi and photo, but also their collaboration. The threshold values are set depend on the size of candidate supporting photos, Argus tends to be aggressive when the size is big. As we can see in Figure 16, our mechanism (denoted by With Photo Selection) obvi-



(a) Shopping Mall (b) Food Plaza

Figure 16. Impact of Photo Selection



(a) Shopping Mall (b) Food Plaza

Figure 17. Handling Device Diversity

ously outperforms the random selection (denoted by Without Photo Selection), especially in terms of large errors.

Handling Device Diversity

Figure 17 plots the results applying our method (Handling Device Diversity) and using raw WiFi fingerprints (Ignoring Device Diversity). As is shown, our scheme successfully and consistently reduces the localization error by about 40%, indicating the severity of not handling device heterogeneity in RSS-based localization.

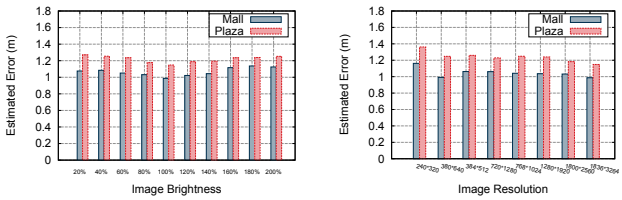
System Overhead

Overall Latency and Decomposition

We evaluate the overall latency by simulating consecutive user queries. We randomly select a photo from the set of query photos and Argus returns the estimated location. Then we average the time delays for 1000 such queries. The time for photo uploading is measured separately by batch. The reason is that we want to measure the time for localization only, without human factors like choosing angle to take photos and checking the localization result. In summary, the file uploading takes 1.8s and image match module consumes 0.6s. The photo selection together with constraint generation spends about 3.3s and the localization algorithm takes another 2.2s. In parallel, the WiFi fingerprint localization considering the device diversity takes 0.5s. So in total that is 7.9s (i.e., $1.8{+}0.6{+}max(0.5, 3.3){+}2.2$). Considering that classic WiFi localization usually takes several samples (e.g., scan 5 WiFi samples in 4.8s [14]), Argus does not introduce much latency than WiFi localization. On the other hand, eliminating the fingerprint ambiguity by motion usually relies on step detection and particle filter [8], where convergence is not guaranteed.

Energy Overhead

We evaluate the energy overhead of Argus using the tools and methodology in [37]. We conduct the experiments using Samsung Note 10.1 tablet, and compare the energy consumption in three scenarios: running nothing, WiFi scanning only, and consecutively using Argus. The frequency of WiFi scanning is $2s$, and the screen is active during the experiment



(a) Impact of Illumination (b) Impact of Resolution

Figure 18. Other Effect Factors

to prevent entering the hibernating mode. The power consumptions for the first two modes are 562mW and 576mW, respectively. Then we consecutively use Argus for localization in groups (10 location queries as a group, and 10 groups are measured here). The average lasting time for a group is $159s$ and a group consumes 116.6 Joules, or 733mW. So Argus consumes 171mW additional power, which is much less than the average power when the screen is active (562mW). Since users only activate Argus when in need and the energy consuming operations are left for server, we believe that such overhead has little impact on mobile device's battery life.

Other Effect Factors

Illumination

Intuitively, the performance of image match is sensitive to lighting condition change. However, according to our experiments, we found that indoor illuminations are relatively stable (since lights are always on). Thus, we simulate illumination change by adjusting the image brightness and contrast. The result is shown in Figure 18a. As is shown, Argus is robust to slight illumination change (80% and 120%).

Image Resolution

To emulate the impact of photo quality, we adjust the level of photographic detail by changing the photo resolution, and repeat the experiment for each resolution level. As shown in Figure 18b, Argus is most accurate with original resolution level, with only a slight performance degeneration with fewer photographic detail.

Human Existence

In fact, the aforementioned $1000+$ images from site survey and 200 images from volunteers were taken without avoiding human existence. To analyse the effect of human existence, we conducted an additional experiment in front of a POI, in which 50 images were taken with human existence and 50 were without it. The key statistics are listed in Table 2. The mean localization errors using Argus were $1.29m$ and $0.96m$ for images with and without human existence, respectively. To reduce computation complexity, our scheme does not involve human detection. Consequently, our photo selection mechanism may select photos both with and without human presence. Thus the above experiment serves as references for the upper and lower bounds of localization errors in practical deployment sites where photos taken may occasionally capture unwanted passengers. In addition, we take an in-depth analysis on the root causes of larger error using photos with human presence. And we conclude that the human existence has an effect on both the WiFi signal strength and the photo quality. As we can see from Table 2, human

Table 2. Impact of Human Existence

	With Human	Without Human
WiFi Positioning Error	$3.44m$	$2.72m$
Argus Positioning Error	$1.29m$	$0.96m$
Average Keypoints	3995	4221
Average Matched Pairs	168	275

existence results in less stable WiFi samples, and WiFi only positioning method suffers from human interference. Besides that, human existence not only decrease the number of photo keypoints, but also the number of matched keypoint pairs between the query image and the supporting images. Since Argus localizes user using coarse WiFi localization and photo generated geometry constraints, the human existence does deteriorate the localization accuracy. However, the degeneration is acceptable as long as the majority of POI is clearly taken.

DISCUSSION

POI Selection. In Argus, we assume that users understand which POIs are likely to be selected by the system or other users. During the experiments, we select apparent physical features like logos, shop entrances, and find that 50 POIs are enough for covering the mall and plaza (similar to the result in [27]). However, users may occasionally shoot a POI not in the database and the database may lack of data during the initial stage. In such cases, Argus invites the user as a volunteer to upload photos for the POI (3 photos are enough for functionating the POI in Argus), or the user can simply choose to be localized by taking a photo of another POI.

Usage Scenario. The design of Argus mainly focuses on large open space (e.g., airports and shopping malls), where most traditional WiFi localization methods provide unsatisfactory performance. Argus may suffer significant performance degradation in boring rooms in office, because we rely on imagery features to extract geometry constraints and it would be difficult for users to select representative POIs inside a boring room. Things may be complicated in campus-like environment. In areas where hallways or small offices dominate, Argus may degenerate to multi-sample WiFi localization. On the other hand, areas where large open space dominates like meeting room and playground are sweet spots for Argus. In addition, Argus relies on established WiFi fingerprint database and does not provide continuous localization when the user is moving. We are seeking the solution to inferring user motion by combining other techniques (e.g., dead-reckoning and particle filter [8, 34]).

Reasons of Large Errors. Though Argus triples the localization accuracy of traditional RSS-based localization method, large errors do exist (e.g., 25 queries have error larger than $5m$ in Overall Localization Accuracy as in Figure 13b). By examining the 25 queries, we summarize the reasons of large errors as: low quality WiFi sample(14), extreme distances or angles(5), similar appearance of POI(4), and obstruction of POI(2). Low quality WiFi samples and query photos lead to large localization error. However, we believe that some large errors can be avoided when either WiFi sample and query photo is in high quality and we plan to improve our photo selection mechanism in future work.

RELATED WORKS

Fingerprint-based Indoor Localization. Due to the ubiquity of WiFi, extensive research efforts utilize it for indoor localization. [5, 36] are pioneers while some recent works take advantage of other smart phone sensors to generate various fingerprints [4, 28]. Recently, researchers have been addressing fingerprint ambiguity by motion-assistance [8, 11, 26, 34] and peer-assistance [14, 21, 23]. Using MIMO [2, 31] is a new trend in indoor localization. The most relevant work to Argus is [14] and [26]. [14] uses peer assist localization and sound ranging to alleviate the WiFi fingerprint ambiguity while [26] utilizes user trajectories to eliminate candidate locations. Differently, we utilize CV techniques to generate geometric constraints using commodity phones without the help of peers and motion trajectory.

Image-based Indoor Localization. Image-based localization is well studied in the robot navigation communities [18], and there has been an increasing research interest in image-based localization with mobile phones [7, 17, 27, 38]. OPS [17] allows users to locate remote objects (e.g., TV tower) by taking a few photos from different known locations, while in Argus, the user takes one photo and localizes himself/herself. Moreover, OPS relies on GPS to provide absolute coordinates, which is infeasible indoors. Sextant [27] localizes users by taking 3 photos from nearby physical features. Different from their triangulation mechanism, Argus localizes users with a single photo by mapping geometric constraints onto the signal space. Jigsaw [7] leverages CV and crowdsourcing to reconstruct the floor plan, based on which it provides convenient localization service. In contrast, Argus works fine without knowing the absolute POI locations or the floor plan.

Crowdsourcing in Localization. A fingerprint database is required for RSS-based indoor localization. Some recent work explored to reduce the laborious efforts to build and maintain such databases. LiFS [33] leverages user motions to construct the fingerprint database and update the database by crowdsourcing. Zee [21] tracks inertial sensors in mobile devices carried by users while simultaneously performing WiFi scans. Similarly, by collecting photos from volunteers and accumulating query images from crowds, Argus is able to work at very beginning and evolve with enriched image database.

CONCLUSION

With the trends towards enhanced wireless connectivity, improved imaging technique, and adoption of the cloud for low-cost, scalable computation, we envision widespread user-friendly indoor location-based service. We developed Argus, an image-assist indoor localization system utilizing geometric constraints from crowdsourced images. Argus makes it possible to localize accurately and efficiently with a single click. The experiments using real data have shown that Argus localizes users in comparable time with classic RSS-based methodologies while triples the localization accuracy.

ACKNOWLEDGEMENT

This work is supported in part by HKRGC under grants GRF-621413 and GRF-16211614, and a Microsoft grant MRA14EG05. It is also supported in part by the NSFC Major Program under grant 61190110 and 61171067, NSFC Distinguished Young Scholars Program under grant 61125202, and Beijing Nova Program under grant Z151100000315090.

REFERENCES

1. `http://maps.google.com/help/maps/indoormaps/faqs.html`. Accessed: 2014-11-25.
2. Adib, F., Kumar, S., Aryan, O., Gollakota, S., and Katabi, D. Interference Alignment by Motion. In *Proc. of ACM MobiCom* (2013).
3. Arkin, E. M., Chew, L. P., Huttenlocher, D. P., Kedem, K., and Mitchell, J. S. An Efficiently Computable Metric for Comparing Polygonal Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 3 (1991), 209–216.
4. Azizyan, M., Constandache, I., and Roy Choudhury, R. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *Proc. of ACM MobiCom* (2009).
5. Bahl, P., and Padmanabhan, V. N. RADAR: An In-building RF-based User Location and Tracking System. In *Proc. of IEEE INFOCOM* (2000).
6. Fang, S.-H., Wang, C.-H., Chiou, S.-M., and Lin, P. Calibration-Free Approaches for Robust Wi-Fi Positioning against Device Diversity: A Performance Comparison. In *Proc. of IEEE VTC* (2012).
7. 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).
8. Hilsenbeck, S., Bobkov, D., Schroth, G., Huitl, R., and Steinbach, E. Graph-based data fusion of pedometer and wifi measurements for mobile indoor positioning. In *Proc. of ACM UbiComp* (2014).
9. Kjærgaard, M. B. Indoor Location Fingerprinting with Heterogeneous Clients. *Elsevier Transactions on Pervasive and Mobile Computing 7*, 1 (2011), 31–43.
10. 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.
11. 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).
12. Li, L., Hu, P., Peng, C., Shen, G., and Zhao, F. Epsilon: A Visible Light Based Positioning System. In *Proc. of USENIX NSDI* (2014).
13. Li, L., Shen, G., Zhao, C., Moscibroda, T., Lin, J.-H., and Zhao, F. Experiencing and Handling the Diversity in Data Density and Environmental Locality in an Indoor Positioning Service. In *Proc. of ACM MobiCom* (2014).
14. Liu, H., Gan, Y., Yang, J., Sidhom, S., Wang, Y., Chen, Y., and Ye, F. Push the limit of wifi based localization for smartphones. In *Proc. of ACM MobiCom* (2012), 305–316.
15. Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *Springer International Journal of Computer Vision 60*, 2 (2004), 91–110.
16. Mahtab Hossain, A., Jin, Y., Soh, W.-S., and Van, H. N. SSD: A Robust RF Location Fingerprint Addressing Mobile Devices' Heterogeneity. *IEEE Transactions on Mobile Computing 12*, 1 (2013), 65–77.
17. 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).
18. Mautz, R., and Tilch, S. Survey of Optical Indoor Positioning Systems. In *Proc. of IPIN* (2011).
19. Park, J.-g., Curtis, D., Teller, S., and Ledlie, J. Implications of Device Diversity for Organic Localization. In *Proc. of IEEE INFOCOM* (2011).
20. Priyantha, N. B., Chakraborty, A., and Balakrishnan, H. The Cricket Location-Support System. In *Proc. of ACM MobiCom* (2000).
21. Rai, A., Chintalapudi, K. K., Padmanabhan, V. N., and Sen, R. Zee: Zero-effort Crowdsourcing for Indoor Localization. In *Proc. of ACM MobiCom* (2012).
22. Sattler, T., Leibe, B., and Kobbelt, L. Fast Image-Based Localization using Direct 2D-to-3D Matching. In *Proc. of IEEE ICCV* (2011).
23. Shen, G., Chen, Z., Zhang, P., Moscibroda, T., and Zhang, Y. Walkie-Markie: Indoor Pathway Mapping Made Easy. In *Proc. of USENIX NSDI* (2013).
24. Snavely, N., Seitz, S. M., and Szeliski, R. Photo Tourism: Exploring Photo Collections in 3D. *ACM Transactions on Graphics 25*, 3 (2006), 835–846.
25. 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.
26. Sun, W., Liu, J., Wu, C., Yang, Z., Zhang, X., and Liu, Y. Moloc: On distinguishing fingerprint twins. In *Proc. of IEEE ICDCS* (2013), 226–235.
27. 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).
28. Wang, H., Sen, S., Elgohary, A., Farid, M., Youssef, M., and Choudhury, R. R. No Need to War-Drive: Unsupervised Indoor Localization. In *Proc. of ACM MobiSys* (2012).
29. Wu, C. Towards Linear-Time Incremental Structure from Motion. In *Proc. of IEEE 3DV* (2013).
30. Xie, H., Gu, T., Tao, X., Ye, H., and Lv, J. MaLoc: A Practical Magnetic Fingerprinting Approach to Indoor Localization using Smartphones. In *Proc. of ACM UbiComp* (2014).
31. Xiong, J., and Jamieson, K. ArrayTrack: A Fine-Grained Indoor Location System. In *Proc. of USENIX NSDI* (2013).

32. 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).
33. Yang, Z., Wu, C., and Liu, Y. Locating in Fingerprint Space: Wireless Indoor Localization with Little Human Intervention. In *Proc. of ACM MobiCom* (2012).
34. Yang, Z., Wu, C., Zhou, Z., Zhang, X., Wang, X., and Liu, Y. Mobility Increases Localizability: A Survey on Wireless Indoor Localization using Inertial Sensors. *ACM Computing Surveys (CSUR) 47*, 3 (2015), 54.
35. Yang, Z., Zhou, Z., and Liu, Y. From RSSI to CSI: Indoor Localization via Channel Response. *ACM Computing Surveys (CSUR) 46*, 2 (2013), 25.
36. Youssef, M., and Agrawala, A. The Horus WLAN Location Determination System. In *Proc. of ACM MobiSys* (2005).
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).
38. 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).