Contextual-Code: Simplifying Information Pulling from Targeted Sources in Physical World

Yang Tian, Kaigui Bian, Guobin Shen*, Xiaochen Liu, Xiaoguang Li, and Thomas Moscibroda*
School of EECS, Peking University, Beijing, China
*Microsoft Research
{tianyangty, bkg, hyperchris, xiaoguangli2010}@pku.edu.cn, *{jackysh, moscitho}@microsoft.com

***Abstract*—The popularity of QR code clearly indicates the strong demand of users to acquire (or pull) further information from interested sources (e.g., a poster) in the physical world. However, existing information pulling practices such as a mobile search or QR code scanning incur heavy user involvement to identify the targeted posters. Meanwhile, businesses (e.g., advertisers) are also interested to learn about the behaviors of potential customers such as where, when, and how users show interests in their offerings. Unfortunately, little such context information are provided by existing information pulling systems. In this paper, we present *Contextual-Code (C-Code)* – an information pulling system that greatly relieves users' efforts in pulling information from targeted posters, and in the meantime provides rich context information of user behavior to businesses. C-Code leverages the rich contextual information captured by the smartphone sensors to automatically disambiguate information sources in *different* contexts. It assigns simple codes (e.g., a character) to sources whose contexts are not discriminating enough. To pull the information from an interested source, users only need to input the simple code shown on the targeted source. Our experiments demonstrate the effectiveness of C-Code design. Users can effectively and uniquely identify targeted information sources with an average accuracy over 90%.**

I. Introduction

In real life, people often want to obtain additional information about a physical object or displayed content, such as learning more about the promotion advertisement on a roadside bulletin board, or a merchandise in a retailer store in comparative shopping scenarios. Conventionally, people may resort to mobile search. Unfortunately, it is hard to find proper search keywords and the search results are typically very noisy [10]. Poised as an easy and precise entrance to the digital world, quick-response (QR) code has become prevalent in the past few years. It is a common practice that a QR code is embedded or attached to a physical object to facilitate the need of additional information [1], [8]. However, despite its prevalence, QR code still demands significant user effort such as moving close enough to the code, holding stable the camera, in addition to requiring good lighting conditions [17].

On the other hand, physical world businesses (e.g., advertisers and merchants) have strong desires to learn more about their potential customers, and in particular, the contexts and access patterns – where, when and how – their ads and goods are accessed. Such contextual information can help them to conduct physical analytics, e.g., to evaluate and improve the effectiveness of existing ads/goods placement [7], [13] and/or even store planning [11]. However, to our knowledge, there is no effective technical solution yet other than user surveys.

In this paper, we present the design of *Contextual Code* (C-Code) – an effective *information pulling* system (IPS) that brings values to both consumers (with significantly simplified information acquisition) and businesses (by providing customers' access patterns) simultaneously. Central to C-Code system is the recognition of the following two facts. First, physical information sources (e.g., a poster) reside in certain real world contexts, and different contexts can be leveraged to differentiate different sources. Second, modern smartphones are equipped with multiple sensors, and they can be exploited to automatically sense the physical world context. With the rich contextual information, we can relieve, and even dismiss, user's efforts in identifying the target information source, and hence significantly simplify user's information pulling experiences.

The challenge arises, however, from the fundamental facts that the contexts of nearby information sources can be very similar, especially when the information sources are densely placed. Given the availability of sensible context features, the sensitivity of sensors and the stability of contextual signals limit the extent to which these features can discriminate different contexts. Therefore, it may end up with multiple candidate sources residing in indistinguishable sensed contexts. However, it may be difficult for a user to screen out the target source from a list of candidate sources that have similar sensed contexts, just as finding the wanted information in mobile search results that contain noise. Hence, the successful identification of the target source depends on how much the list of candidate sources can be shortened.

To solve this problem, C-Code involves the user in the loop and leverages user's recognition capability to help disambiguate the target source from those in the similar context. As the usage of QR code, C-Code system also assigns different codes to the sources whose contexts confuses with each other. When a user wants to pull information from a particular source, she indicates her intention by inputting the code seen on the source. The context information automatically sensed by the phone and the code input by the user will *jointly and precisely* identify the target information source, hence the name Contextual Code. C-Code system can easily fulfill its value proposition to the businesses by deriving the users' access patterns from the acquired context information.

Involving the user in the loop, C-Code seeks to *maximally reduce the user's manual effort*. Ideally, the code should be easy to discover, to remember and easy to input or reproduce by the user. Our design goal is to let user simply type in a single digit/character or input a phone gesture (e.g., shaking the phone). It is obvious that the more we can distinguish among contexts, the simpler code we can use. To further improve user experiences, we want to retain a small, fixed code dictionary so that we can present the codes as buttons on phone screen and users only need to tap on big buttons.[1] To this end, we need to reuse the code as much as possible.

Based on these considerations, in this paper, we focus on the following three technical problems: 1) Context sensing that aims to maximally discriminate contexts using signals of complementary sensor modalities; 2) Code assignment that

[1] The user interface is shown in Figure 7.

minimizes the code space and the conflict between codes assigned to information sources in the same context; 3) C-Code matching that maps a context signal to a known context, given possible signal drifts of certain sensor modalities.

To maximally discriminate contexts, we leverage multiple sensor modalities to increase the dimensionality of a context, and represent the signals of each context feature as a certain probability distribution. Second, we establish a unified probabilistic framework for analyzing the interference between two contexts, formulate a code assignment problem using the interference graph, and solve the problem with an effective heuristic algorithm. Thirdly, we devise a probabilistic matching algorithm that combines multiple complementary sensor modalities to achieve a high matching accuracy, together with an implicit user feedback mechanism that exploits the natural user inputs to organically grow the set of context signals to be drift-resistant.

We have implemented C-Code client and server, and evaluated it in the office buildings. Our experimental results show that the C-Code system can achieve the mapping accuracy of over 90% precision and recall rates, by combining all of the context features, the user-input code, and the user feedbacks.

As a remark, we point it out upfront that context in C-Code is more than a *location* and the location, if available, can be a component of a context. For example, two posters face-to-face posted on two opposite walls at the *same* location are considered to reside in two *different* contexts. Not interested in figuring out the actual location of the information source or the user, C-Code explores contexts *without* going through a localization process. Hence, it does not rely on the availability of a localization system. However, the exploration of contexts may raise concerns on privacy. In C-Code, the context information are sanitized not to contain any personally identifiable information. How to further protect user's privacy is out of scope of this paper.

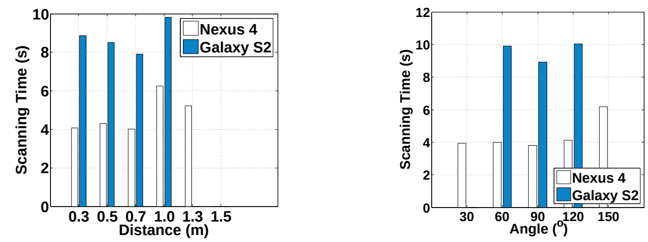
II. Background

Users' Desire of Information Pulling: There is often a need for users to pull further information from an information display, be it a roadside ads/poster or a merchandise. For example, a user wants to learn more about the promotion seen at the entrance of a mall; a user may want to see the reviews and compare prices of a particular object when shopping. To fulfill such information pulling demand, people may resort to mobile search. However, it requires a user to find proper keywords, which is proven hard [10]. The search results are also generally noisy, and is painful to screen out the actual wanted information. Worse even, current search engines have not reached the granularity of indexing physical objects. As a result, mobile search can only answer queries of coarse level physical entities and the general knowledge.

To facilitate users' desire of additional information, user-input codes (a phone number, a URL, or a scannable QR code) are often embedded on an ads/poster, through which a user can retrieve precise information about the ads/poster. These user-input codes are typically lengthy to uniquely identify the information source, which unfortunately sacrifice user friendliness. It is hard to remember and to input a long string, be it a URL or a phone number. Poised as a solution to simplify user input, QR code has become prevalent.

Businesses' Desire of User Contexts: User's desire of effective information pulling is only half of the story in the overall information dissemination and acquisition system. Physical world advertisers and merchants also have a strong desire to learn more about their potential customers, and in particular, the contexts and access patterns – where, when and how – their advertisement and goods are accessed. Such contextual information can help them to conduct physical analytics, e.g., to evaluate and improve the effectiveness of existing ads/goods placement [7], [13] and/or even store planning [11]. However, to our knowledge, there is no effective solution yet. The effectiveness of ads placement is typically evaluated via user surveys. Physical analytics is emerging and would require a localization system in place [9], [12], which is unfortunately still far from a reality [15], [21], [23], [24].

Experiences with QR Code: Despite the popularity of QR code, it is still effort-taking and faces environmental constraints [17]. For example, the successful scan of a QR code requires the user to be in a close proximity to the code, hold steady the camera, as well as good angles of scanning.



(a) Scanning distance; (b) Scanning angle.

Fig. 1. The scanning time of QR codes under various conditions on different devices.

We perform experiments to gain a sense of the QR code scanning experience. We use the time it takes to successfully scan a QR code as a metric. We printed two 0.1 m×0.1 m QR codes on a paper, containing messages of 108 bytes. We used two QR code scanning app ZXing on two Android devices (Glaxy S2 and Nexus 4) for the experiments.

Figure 1(a) shows the impacts of scanning distance: the effective range is indeed very small, from 0.3m to 1.5m. For other experiments, we fix the phone's camera at a distance of 0.3m which is the best working distance. Figure 1(b) shows the impact of scanning angle. Clearly, the success of scanning a QR code depends on the proximity to the code and the scanning angle. The working distance of a C-Code is usually much larger than that of a QR code, which varies in different environments and depends on the distance between neighboring C-Codes that have the same user input code but distinguishable sensed contexts.

The close proximity requirement and effort-taking experience may also readily expose user's intention and may discourage the users from scanning the QR code in public. For example, a user may choose not to scan in public a QR code on a poster promoting adult contents even if he actually wants to learn more.

III. C-Code System Overview

We want to improve the user experiences by minimizing user's efforts while being able to precisely identify an information source. To this end, we propose the *Contextual-Code* system. In this section, we give an overview of the system.

A. Contextual-Code Concept

Every physical information source resides in a certain context, which contains a number of context features endowed by the environment, including location, radio environment (WiFi, cellular, BlueTooth), magnetic field, lighting, etc. Modern smartphones are equipped with multiple sensors (e.g., GPS, cellular, WiFi [3], sound [19], magnetometer [6], light sensor [2], etc.) that can be exploited to sense these context features.

The core idea of the proposed C-Code system is to leverage the context information automatically captured by the smartphone to help distinguishing different information sources in different contexts. Given the availability of sensible context features, the sensitivity of sensors and the stability of contextual signals limit the extent to which these features can discriminate different contexts. To work around, C-Code resorts to the user's help to further differentiate sources under indistinguishable sensed context.

Context: Formally, we define a *context* as a *physical proximity* to an information source in which a user can freely access the source. Clearly, a context here is not a rigid space, but subject to user natural sense of access. For example, a user may feel a poster is accessible when facing it within a few meters instead of backing to it or at tens of meters away.

Note that context is more than a *location*. For example, two posters face-to-face posted on two opposite walls at the same location are considered to reside in two different contexts. In our target scenarios, we are not interested in figuring out the actual location of the user, even though many context features are locality-preserving and can be used for localization purpose. Our use of context does not go through a localization process. Hence, C-Code does not rely on any localization system.

Contextual-Code: A Contextual-Code (C-Code) is defined as a two-tuple $(\mathbf{s}, c)$, where $\mathbf{s}$ denotes the signal space of all features observed (by mobile phone sensors) in the context where an information source reside, c denotes the *user-input code* attached to the source to help discriminate different sources. Note $\mathbf{s}$ or c may not be unique, but their combination, i.e., the C-Code, should be unique to precisely identify a particular target.

In practice, different information sources may in the similar contexts have the same code, due to inappropriate code assignment or practical issues in deployment. We refer to such a case as a *C-Code conflict* between the two sources. In such cases, a C-Code cannot uniquely identify any of the two sources because their contexts cannot be well disambiguated.

Note that it is possible to acquire context information but stick to existing QR code. Such a design would be, however, an overkill to uniquely identify the information source, and still incur substantial user efforts of QR code scanning.

Scope of the Paper: The targeted information sources may appear in stationary or dynamic contexts, such as ads on billboards or on buses). Meanwhile, the sources may contain static or dynamic content, such as posters on paper media or multimedia ads in public display. In this paper, we focus on the "stationary" contexts and information sources with "static" content.

B. Architecture and Operation

System Architecture: The architecture of the C-Code system is shown in Figure 2. It involves three logical parties, namely content provider, service provider and end user (i.e., information puller). The *content provider* is an entity that maintains a knowledge base containing the detailed information/content of the information sources. The *service provider* is an entity that provides publicity services to content provider and provides access methods to users to access their intended knowledge base. The service provider collects the context information of all information sources (in an offline phase), assesses the discernibility among collected contexts, and assigns different codes to information sources whose contexts confuse with each other. The contexts and their associated codes are stored in a database with additional links to related knowledge base. Upon receiving a C-Code, it matches against the database and return the corresponding links to the user. The *information puller* makes inquiries for specific interested information source. To do so, the user's phone automatically senses contextual information, couples them with user input code, and sends the resulting C-Code to the server. It also displays returned links to the user and may implement some feedback mechanism.

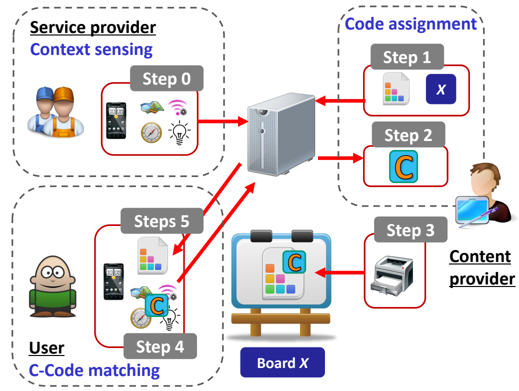


Fig. 2. The C-Code system architecture.

Operation Flow: Figure 2 also shows the operation flow of the C-Code system. First, the service provider collects offline the context signals for information sources using a smartphone (Step 0). Then, the content provider registers with the service provider and selects sources to publish their ads/poster (Step 1), the C-Code server assigns user-input code for the selected contexts and establishes the association among the contexts, and the code and the ads/poster (Step 2). Next, the content provider embeds the code or attaches the printed code to the ads/poster, and publishes in the selected sources (Step 3). An end user that wishes to pull information of the sources will recognize the user-input code on the targeted source, input the code on the phone interface (Step 4), and retrieve additional information about the targeted source (Step 5).

C. Key Challenges

Our goal is to minimize user's effort in pulling information from a specific information source. We design to use contextual information to distinguish different sources and thus to reduce user effort. There are a few technical challenges towards this goal. First, it is obvious that the more we can distinguish two contexts, the simpler code we can use. Thus, the primary challenge is how to choose the contexts features for maximally differentiating two contexts. In addition, we need to effectively match the context signals from the end user against those collected by the service provider. Second, for better user experiences, we hope to retain a small, fixed code dictionary so that we can present the codes as buttons on phone screen and users only need to tap on big buttons instead of typing in. To this end, we need to reuse the code as much as possible. Thus, the second challenge is how to minimize the code space (the number of codes), or for a fixed code space, to minimize the probability of C-Code conflicts – same code are assigned to multiple sources in indiscernible contexts.[2] Thirdly, it is

[2]C-Code conflicts may happen in two cases: 1) improper assignment of the user input code c due to code space limit; 2) different sensing capability of user's smartphones may lead to different discernibility of $\mathbf{s}$ as those assumed in code assignment.

undesirable for the service provider to repeat collecting the signals of time-variant contexts for all information sources. Thus, the system needs to dynamically adapt to contexts by organically updating the context signals in the context matching procedure.

IV. Multimodal Context Sensing

In the context sensing mechanism, the service provider has to complete two tasks: (1) choose complementary sensor modalities that can maximally distinguish two contexts, and (2) transform the collected signals of each context feature to a probability distribution or a range-based representation.

A. Context Feature Selection

Intuitively, the more context features we use in C-Code, the better two contexts will be distinguished. However, our research findings show that only a few complementary context features will suffice instead of using all of the features.

We first define a set of *common* context features that can be easily captured by sensors of ordinary smartphones, including GPS, cellular, WiFi, magnetic field, sound, and light. Note that barometer is not available on ordinary smartphones, and the atmospheric pressure is not a candidate in our consideration.

Then, we choose a few *candidate* context features from the set of common features as follows. We found GPS and cellular are overlapped with WiFi in terms of representing the location information, and thus we use WiFi as the representative of GPS and cellular. Moreover, the readings (in lumens) of light sensors in an environment heavily depend on the sunlight that may vary according to the local time and weather. As the weather is not a context feature that can be sensed and quantified, we remove "light" from being used as a candidate in the C-Code. As a result, we have three candidate context features, namely WiFi, sound and magnetic field.

Next, we experimented to observe why the selected candidate context features are complementary and how much we can discriminate two contexts using these candidates. We revisited the definition of *similarity* between two locations using context features such as WiFi [2], sound [19], and magnetic field [26]. We call two contexts are similar regarding a context feature, if they have a high similarity for the given feature. We carried out experiments in an office building where face-to-face posters are placed on two sides of walls along the corridor. We choose 16 locations separated by 2 m along the corridor as contexts. We are able to make the following two observations.

A) Context features have different capabilities of disambiguating contexts: We show the similarity between two contexts for WiFi in Figure 3 by using four types of devices, and we observe that WiFi signals for two distant contexts have a low similarity, and thus can be well disambiguated. Figure 4 shows the sound signals of three chosen contexts. Context w is close to an air conditioner and it leads to different observations from the other two contexts. Due to the dependency on the ambient background, the sound signals for two contexts distant to the sound source may have a high similarity; and it is possible to distinguish contexts that are at different distances to the sound source.

B) Context features are complementary: We choose 10 locations separated by 2 m and point the phone to face-to-face posters on two sides of walls at each location. Each location is then transformed to two different contexts because of different facing directions detected by the phone magnetometer. In Figure 5, we plot the similarity heatmaps for each single feature of WiFi, sound, magnetic field, as well as for the simple weighted average of the three. Clearly, using three features is better than a single feature for discriminating two contexts as they have a low similarity value in the fourth heatmap.

Based on these observations, the sensor modalities we chosen are independent and complementary. Specifically, sound or magnetic signal (the ambient background-dependent feature) is locally more discriminative but not globally, where as WiFi (the IT infrastructure-based feature) is globally discriminative but not locally.

B. Context Representation

In this paper, we focus on context features that are observable to a smartphone, and let $F = \{f_1, ..., f_i, ..., f_k\}$ denotes the set of these observable context features, where k represents the number of context features in consideration. A context feature can be scalar or vector. For example, sound at a given frequency is a scalar feature whose signal is in a value in units of db; WiFi fingerprint is a vector feature consisting of components—i.e., received signal strengths (RSSes) from a set of WiFi APs. Suppose feature i is a vector that contains κ_i components, and we simply call feature i and its component j ($j \in [1, \kappa_i]$) as feature i, j.

As aforementioned, a context refers to a range of space proximity to a source. Therefore, the observed signal for each context feature (a scalar feature or a component of a vector feature) can span a range of values as signals may be collected from multiple spots in the same context. In fact, due to time variation of signals, multiple collections of context signal at the same location also spans a range of values.

To facilitate more effective representation, we choose to model the signal distribution for each scalar feature and component of a vector feature. We can observe that the signals of each feature can span a range of values, and approximately follow a certain distribution. We are able to draw the best-fit Gaussian distribution from the histogram of the collected signal samples. In Figure 6, we choose two example contexts u and v, and show their histograms of percentage vs. signal RSS for WiFi and sound features. In this example, the ranges of WiFi signals have an overlap, while the ranges of sound signals overlap little. This implies that both contexts u and v are pretty close to certain WiFi APs (with RSS over -60 dBm), but one of them is much closer to a sound source than the other. It also implies that using two complementary features will better distinguish two contexts than using a single feature, say WiFi.

More specifically, given a context u, we represent the signals for feature f_i using a Gaussian distribution for the features we selected, i.e., sound, magnetic field, and WiFi. The signal for a given feature f_i, component j under context u, is denoted as $s^u_{i,j}$ ($j = 0$ if feature i is a scalar feature), which can be represented as $N(m(s^u_{i,j}), \sigma(s^u_{i,j}))$ where $m(s^u_{i,j})$ is the mean value of collected signal samples, $\sigma(s^u_{i,j})$ is the standard deviation.

To mitigate possible outliers in sensed signals, we further constrain the data range to be

$$[m(s^u_{i,j}) - \delta(s^u_{i,j}),\ m(s^u_{i,j}) + \delta(s^u_{i,j})]$$

where $\delta_i(s^u_{i,j}) = 3\sigma(s^u_{i,j})$ means about 99.7% of values drawn from a Gaussian distribution are within three standard deviations away from the mean. We call $\delta_i(s^u_{i,j})$ as half of the *data range* of the signals for feature i, j.

V. Code Assignment

The code assignment is based on the use of an interference graph where each vertex represents an information source, and an edge has a weight that quantifies the interference between

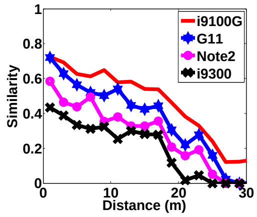
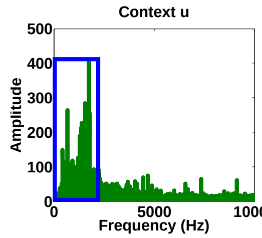
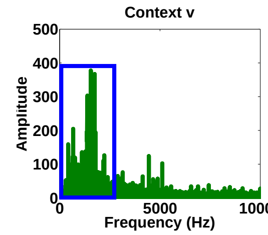


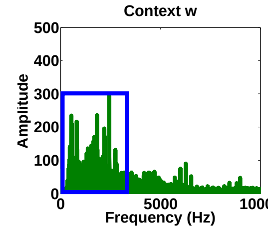
Fig. 3. The similarity between contexts for WiFi vs. context separation distance.



(a) Context u distant to the sound source;

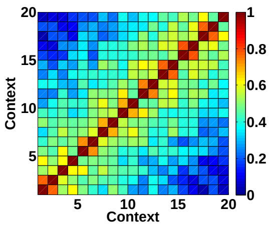


(b) Context v distant to the sound source;

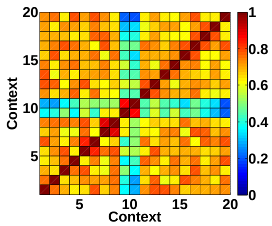


(c) Context w close to the sound source.

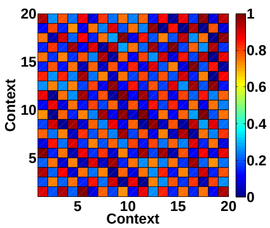
Fig. 4. Observed sound signals in three chosen contexts.



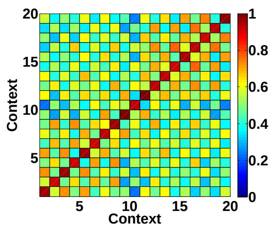
(a) WiFi;



(b) Sound;



(c) Magnetic field;



(d) All three combined.

Fig. 5. Heatmaps of similarity for three context features at contexts of ten chosen locations, where two locations are separated by 2 m.

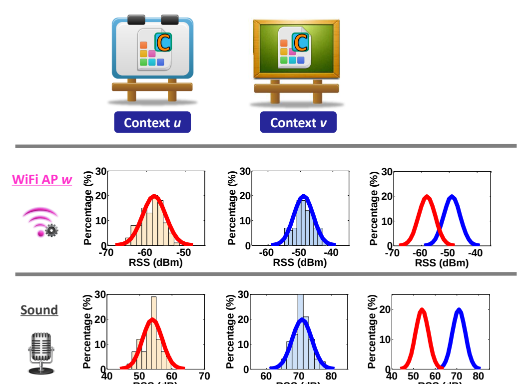


Fig. 6. Illustration of the observed context feature signals as a range: comparing two contexts u and v using signal histograms for WiFi and sound features.

the two neighboring contexts. Before constructing the graph, we first estimate the interference relationship between two contexts.

A. Interference Likelihood Estimation

Let $I(s^u_{i,j}, s^v_{i,j})$ denote the interference likelihood between two signals $s^u_{i,j}$ and $s^v_{i,j}$ for feature i, j, under two contexts u, v. As Figure 6 shows, the distributions of a given feature for two contexts may overlap. Let $[\alpha_{i,j}, \beta_{i,j}]$ denote the overlapping part of their data ranges. We use $s^* \in [\alpha_{i,j}, \beta_{i,j}]$ to denote a pivot signal value such that

$$P(s^u_{i,j} = s^*) = P(s^v_{i,j} = s^*).$$

Without loss of generality, we assume

$$m(s^u_{i,j}) - \delta(s^u_{i,j}) < m(s^v_{i,j}) - \delta(s^v_{i,j}),$$

That is, the distribution of one context cannot be a subset of the other. Thus, the overlapped data ranges can be further divided into $[\alpha_{i,j}, s^*]$ and $[s^*, \beta_{i,j}]$.

We next define the interference likelihood between two contexts u, v, for feature i, j as the overlapped area between their distribution histograms divided by two:

$$I(s^u_{i,j}, s^v_{i,j}) = \frac{\int_{\alpha_{i,j}}^{s^*} p(s^u_{i,j})ds + \int_{s^*}^{\beta_{i,j}} p(s^v_{i,j})ds}{2},$$

where $p(s^u_{i,j})$ denotes the probability density function for signal $s^u_{i,j}$.

We calculate the interference between two contexts u and v for all features i, j in F as

$$\mathcal{I}(u, v) = \min_{\forall i,j}\{I(s^u_{i,j}, s^v_{i,j})\}. \quad (1)$$

Note that in Equation (1), we use the *minimum* interference likelihood among all context features to represent the interference between two contexts. Based on the data ranges for all features of two contexts, we see that the single feature that leads to the smallest interference likelihood is the most discriminative one to maximally distinguish the two contexts. In addition, we have based our design on a set of sensor modalities that modern smartphones are commonly equipped with. A legacy, less sensing-capable device will suffer from more conflicts, and hence noisier results. If new context features can be commonly sensed by future generations of mobile phones, they can be easily incorporated into C-Code using Equation (1).

We call two contexts u and v interfere with each other if $\mathcal{I}(u, v) \geq \tau$, where τ is the interference severity threshold. In other words, the two interfering contexts have significant similarity in context features, and sources in them may not be well distinguished.

B. Code Assignment Problem

The code assignment problem we are facing has two-fold goals. Firstly, we want to minimize the overall, system-wide code space. Secondly, when a constraint on the code space size is imposed, we want to minimize the resulting C-Code conflicts. The problem is very similar to the channel assignment problem in cellular or WiFi networks, where transmissions of two neighboring/adjacent network base stations will collide with each other when they are assigned the same channel.

Context Interference Graph: Channel assignment in WiFi networks is usually solved via a network interference graph, where the edge between two vertices indicates the interference relationship between two network base stations. A transmission conflict will occur when two connected vertices in the graph are assigned with the same channel [16].

We borrow the idea of network interference graph and solve the code assignment problem by constructing a *context interference graph* $G = (V, E)$, where the vertex $x \in V$ represents the context of a registered source x. The C-Code system collects context signals from all its information sources, and assess the interference likelihood among them. Suppose two contexts u, v, and an edge $(u, v) \in E$ has a weight $w(u, v) = \mathcal{I}(u, v) \in [0, 1]$. Obviously, if $w(u, v) \geq \tau$, then information sources at different context u and v will conflict with each other if they are assigned with the same user-input code.

Problem Formulation: When code c_x is assigned to an information source x registered in context u, we use (c_x, u) to represent an assignment decision. Let A be a code assignment to G, which is a set of assignment decisions. Upon a new assignment decision (c_x, u) is made, the assignment A is updated to be $A \cup \{(c_x, u)\}$.

Let U denote the universal set of codes, and C_A be the set of codes used in the assignment A. We call A is a complete assignment if every poster $x \in V$ has received an assignment decision. Otherwise, it is an incomplete assignment. We define *interference degree* in a context interference graph G as the number of interfering pairs of contexts, given a code assignment A, and denote it as $D(G, A)$.

The code assignment problem is equivalent to the graph (vertex) coloring problem. In a graph coloring problem, any pair of two interfering vertices in the graph should be assigned different colors to avoid the interference. Let $V(A)$ denote the set of vertices (contexts) that have received an assignment decision from the assignment A. Suppose the network has an initial assignment A_0 with code set C_0 (it is possible that $A_0 = \emptyset$).

In the ideal case, U could be an infinite set, every content provider follows the C-Code deployment etiquette, and we can always assign different codes to any pair of interfering vertices in the context interference graph. The code assignment problem is formulated as the following one that minimizes the code space in use:

$$\begin{aligned} \min \quad & |C_A| - |C_{A_0}|, \\ \text{s.t.,} \quad & V(A) = V;\ D(G, A) = 0; \\ & c_x \neq c_y,\ \text{if } w(x, y) \geq \tau, \end{aligned}$$

where x and y are two information sources registered with contexts u and v. The constraint implies that A is a complete assignment; the second constraint means that the interference degree in the assignment A should always be free of interference when U could be an infinite set; the third one requires two sources in interfering contexts should be assigned different codes.

In practice, U could be a finite set. If the codes in U are exhausted, i.e., $U \setminus C_A = \emptyset$, there is no way to avoid reusing the code in C_A. Thus, the objective of C-Code system is to minimize the interference degree. Given $G = (V, E)$, the code assignment problem is then formulated as follows:

$$\begin{aligned} \min \quad & D(G, A) - D(G, A_0), \\ \text{s.t.,} \quad & V(A) = V; \end{aligned}$$

C. Code Assignment Algorithm

We propose a C-Code assignment algorithm that is similar to the graph coloring algorithms presented in [16], and is executed in a greedy fashion. Given a context interference graph G with an initial assignment A_0—the un-assigned poster registered in the context with the maximum number of interfering neighboring vertices in G is the next to be assigned a code. When there is no unused code (i.e., $U \setminus C_A = \emptyset$), the code

$$c^* = \arg\min_{c \in C_A} \{D(G, A \cup \{(c, u)\}) - D(G, A_0)\}$$

that leads to the minimum increase for interference degree will be assigned. When all posters registered with all contexts are assigned, the algorithm terminates. Whenever a new poster is registered with a context, the C-Code server incrementally assign codes to the newly-added one. Note that, the content published in C-Code system may expire, e.g., end date of an event. Hence, we can set the expiration time for a code assignment, typically the same as the expiration time of the content. Once it expires, the code can be reclaimed and reused.

VI. C-Code Matching

In the C-Code matching algorithm, the context signal captured by the user's phone is called the *test* signal; and the contexts on C-Code server whose signals are represented as the data ranges are called the *training* contexts.

Recall that the code assignment mechanism employs an interference estimation process, which is a range-to-range comparison between the data ranges of feature signals for two posters' contexts to determine whether they interfere with each other. Moreover, the C-Code system knows the most influential feature for distinguishing two contexts (according to Equation 1).

Unlike interference estimation, the C-Code matching is a signal-to-range comparison to map the test signal to a training context that has a range of signal values for each feature. The C-Code system has no knowledge about which training context the user's targeted poster resides, and which feature is the most effective to identify the context.

Thus, the system cannot exclude any feature from being used in C-Code matching; instead, it probabilistically involves every feature in the matching process. The C-Code system computes the likelihood that a test signal matches a training context signal for each feature, and then normalize these likelihoods given that all training context signals are equally likely to be matched. C-Code then returns the top-matched training context that has the highest matching probability.

Sequential matching as benchmark: We compare the proposed probabilistic matching algorithm with a benchmark algorithm, called the sequential matching algorithm. The benchmark algorithm is executed in multiple iterations sequentially. In each iteration, one context feature i, j is chosen to screen out a set of training contexts whose signals are best matched against the test signal. The algorithm terminates until all context features have been tried.

Probabilistic Matching: Using Bayes' rule, C-Code system computes the probability that a test signal s matches the signal of training context u

$$P(u|s) = \frac{P(s|u) \times P(u)}{P(s)}. \qquad (2)$$

Without prior information about the exact test signal's context, C-Code assumes that the targeted poster is equally likely to be at any context on the server, i.e., $P(u) = P(v), \forall u, v \in V$. Thus, Equation (2) is rewritten as $P(u|s) = c \times P(s|u)$, where c is a constant.

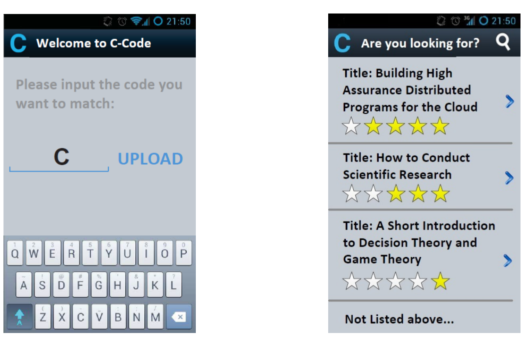
Let $P(s|u)$ denote the matching probability that a given test signal s belongs to context u, and we want to find the context $u^* \in V$ that maximizes the probability of $P(s|u)$, i.e.,

$$u^* = \arg\max_{u \in V}(P(s|u)).$$

The probability $P(s|u)$ can be calculated on basis of the matching probabilities for every feature i, j between the test signal s and context u, such as $P(s|u) = \prod_{i,j} p_{i,j}(s|u)$, where $p_{i,j}(s|u) \in [0, 1]$ is the matching probability for feature i, j between the test signal s and context u. The matching probabilities $(p_{i,j}(s|u))$ can be derived using the non-parametric distribution histogram obtained during the context sensing phase.

Context adaptation: C-Code can return multiple top-matched results according the decreasing order of their matching probabilities. Every user has an option to confirm whether the server has made a correct matching decision: (1) the user confirms it is a correct matching by tapping one of the returned results; or (2) the user confirms there is no correct matched results by tapping "not listed above".

Benefits of user feedbacks: When the user confirms it is a correct matching, the context signals for this match will be added into the server database for updating training context signals. In other words, the context signals get an organic growth/update over time. When the user confirms there is no correct matched result, the current context fails to match any context on the server, and this will not update the training contexts. With the user confirmations, the server knows about the matching correctness; and the organic growth/update of context signals can reduce the matching error introduced by time variance in context signals. We will show the performance gains with organic growth/update of context signals in evaluations.

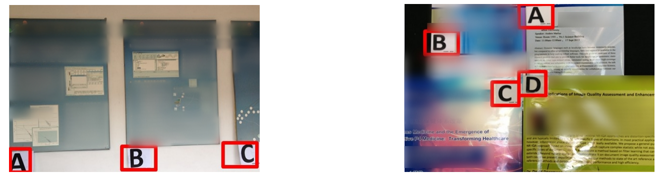


(a) Code input on a keypad; (b) The user's confirmation interface;

Fig. 7. Screenshots of user interfaces.

VII. Implementation and Evaluation

We developed the C-Code client as an Android application, which can (1) automatically upload the collected context signals to the server for C-Code matching by pressing a button on a keypad (Figure 7(a)); (2) display a list of matched results to the user (Figure 7(b)). The server allows a content provider to choose the contexts to publish its posts. It assigns codes to information sources that are registered in a managed context, according to the algorithm described in Section V. It accepts the user-uploaded C-Code and maps it to one matched or multiple candidate information sources via a probabilistic C-Code matching algorithm. For real-time interactions, it sends the matched results back to client and collects user feedbacks to improve the system performance.



(a) Display panels; (b) Poster board;

Fig. 8. Experimental environments.

A. Experimental methodology

Experimental environments: We deployed the C-Code system in two environments on campus. *Display panels* are placed face-to-face on both walls of the building corridor. Three panels form a group (Figure 8(a)), and two groups are separated with an interval of 2 m. We chose four groups on each wall as the managed contexts in experiments. Panels have the same size of $1.5m * 0.9m$, and two panels in the same group are separated with an interval of $0.2m$. Each panel contains only one poster. Two *poster boards* are placed at the door of the office building (Figure 8(b)) with multiple posters, advertisements, announcements coexisting on the same board. The number of posters on each board is typically 4 or 5.

Popularizing C-Code on campus: We manually collected the signals of contexts at display panels, poster boards by conducting two site surveys with an interval of one month, and each site survey consists of three days. Right after the second site survey, we released the C-Code system to 100 volunteer users to use for one month. As a result, we collected three sets of context signals: the set in the first site survey, the set in the second site survey, and the set of data accumulated by users feedbacks in the release period. During the release period, we received 2240 information pulling requests, and every day we received about 70 requests from different users: 31 requests from display panels, 39 requests from poster boards, on average. We assigned four types of devices to the users: Galaxy i9100g, Note2, i9300, and HTC G11. In order to attract users' attentions, we make the simple code (one-digit letter) differentiable from other poster contents by posting it at easy-to-notice places on posters.

Performance metrics: We use precision and recall rates to evaluate the system performance. In C-Code system, high recall means that server returns most of the relevant results, while high precision means that it returns substantially more relevant results than irrelevant. In the experiment, the ground truth for C-Code matching is known because we require volunteers to send user confirmations to the server.

B. Main Results

First, we evaluate the overall performance of C-Code in all environments. By default, we use the data collected in the second site survey as the set of training contexts, and the user feedback mechanism is enabled.

Matching accuracy when varying the number of features: We evaluate the performance of the probabilistic matching algorithm by varying the numbers of used features (labeled with "Prob" in figure legend). Meanwhile, we use the sequential matching algorithm as the benchmark (labeled with "Seq" in figure legend).

In Figure 9, the first group bars with label "GSM" show the results when only one feature GSM is used; each of other groups has a feature added, e.g., "+WiFi" groups means the WiFi feature is added for matching. The results show that the WiFi feature is the most significant contributor in improving the matching accuracy, while each of other features has less significant contribution in differentiating contexts.

The impact of number of returned results: Figure 10 shows the matching accuracy that the list of returned results contains the user's target. The recall/precision rates is no less than 0.9 when the number of returned results is no smaller than three.

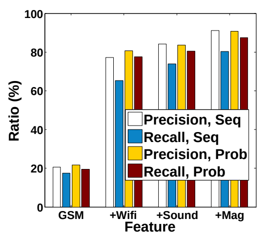


Fig. 9. Varying # of features used.

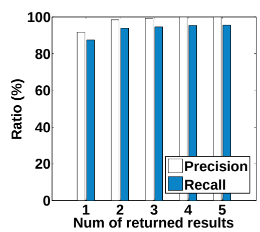


Fig. 10. Varying # of returned results.

Matching accuracy with different sets of training contexts: We show the impact of different sets of training contexts on the matching accuracy in Figure 11. The label "1+2" on the x-axis represents that the set of training contexts includes the data collected in both site surveys; "2" includes the data collected in only the second site survey; "2+" means that the training data set in the second site survey keeps receiving organic updates from the user feedbacks collected in the release period; the fourth group labeled with "release" represents that the set of training context signals only include the data collected by user feedbacks in the release period.

The discrepancy between the first, second, and third groups of bars shows the accuracy improvement when we retire the old data collected in July and receives new organic updates of data collected from the user feedbacks in September. Using the data in the most recent month cannot significantly improve the accuracy (see the "release" group of bars).

Figure 12 shows the change of matching accuracy in each day during the release period. Without the user feedbacks, only the "2" set is used, the recall rate keeps changing at a low level; the precision rate drops a little as the data is getting old as time evolves. When user feedbacks are available, the "2+" set is used, and the recall rate keeps increasing owing to the updated signals provided by user feedbacks.

The impact of organic growth of context signals: Without the signal update from the user feedback mechanism, the probabilistic matching has a better performance than the benchmark in recall and precision rates (Figure 13(a)). The reason is that the error in the sequential algorithm could accumulate from a single feature's matching results to the next one's results. When the set of training contexts collected in the second site survey has no updates from the user feedbacks, it is

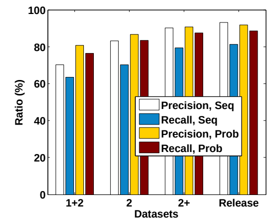


Fig. 11. Retiring old data improves the matching accuracy.

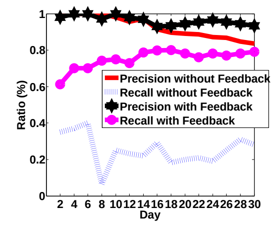
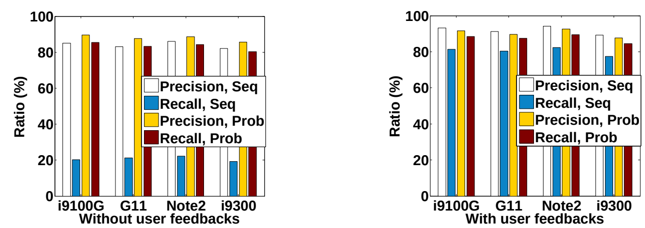


Fig. 12. Probabilistic matching using organic updates.

insufficient to completely encompass all signals' variances. As a result, the sequential algorithm may mistakenly filter out many unmatched contexts regarding a single feature. In contrast, the probabilistic algorithm does not completely exclude unmatched contexts for a single feature, which leads to a better matching accuracy.

When the user feedback mechanism is enabled, we observe that both matching algorithms achieve a high accuracy with the organic update of signals, especially that the recall rate of the sequential algorithm gets a significant boost.



(a) Without user feedbacks; (b) With user feedbacks.

Fig. 13. Improved matching accuracy via user feedbacks.

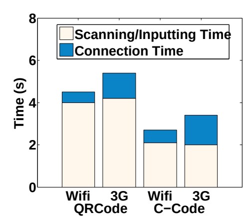


Fig. 14. Time cost of information pulling.

IPS	QR code	C-Code
Display	2.06	1.648
Wifi	0	0.31
Camera	0.42	0

TABLE I. Energy consumption (J) for one-time information pulling.

The device heterogeneity: Different iOS mobile devices have little heterogeneity in the sensor hardware, while Android devices have diversities in sensor hardware. Four tested devices in our experiments have a small variance in matching accuracy as shown in Figure 13. To reduce the impact of device heterogeneity, the service provider may need to maintain a set of signal samples for each type of devices, and each set will be used serving the specific type of devices.

Time and energy cost of using C-Code: Suppose the QR-Code contains a URL to a poster, and the C-Code directs to the C-Code server that has the same poster. Figure 14 shows that, a user only needs about 2 seconds to recognize and input a code on phone, less than the time for pointing the phone to a QR code and then scanning.

Energy consumption is not a concern for either QR code or C-Code system, because people only use these applications sporadically. The major energy consumption for QR code scan is from camera-viewing (about 4 seconds for each QR code scan). For each C-Code input, the major energy consumption is from context sensing by phone sensors completed within two seconds. Table I summarizes the energy consumption for major parts.

VIII. Related work

Although we focus on effective information pulling techniques, C-Code is essentially a system finishing functions of context sensing, code assignment and code matching, which could be treated as a new method of gathering and analyzing information in the physical world [11].

C-Code typically relies on contexts detected by smartphone sensors. GPS [20] has been a satellite-based navigation system, applied in military, civilian, and commercial purposes, however, it shows bad performance for indoor environments. Many other works are designed for improving accuracy of indoor localization. Taking advantage of wide signal-strength fingerprints, GSM [14] indoor localization system could achieve median accuracy of 5 meters in large multi-floor buildings. Meanwhile, a vast majority of existing research efforts depending on RF signatures from certain IT infrastructure come up with new solutions for indoor localization [3], [4], [22], [25]. Jaewoo [6] presents an indoor positioning system that measures location using disturbances of the Earth's magnetic field caused by structural steel elements in a building. Stephen [19] uses Batphone to show the benefits of using acoustic background spectrum together with a commercial WiFi-based localization method. Though these papers make fundamental contributions to localization, contexts mean more than that. Contexts supply more high-dimension information than location, which has proved many novel systems [18], [26]. Moreover, accurate location does not have significant meanings all the time. Contexts may have more advantages than simple coordinates in characterizing ambient environments and providing services [5].

IX. Conclusions

This paper presents an information pulling system, called Contextual-Code (C-Code), that can greatly simplify the user efforts and automatically capture the user's context information for the purpose of physical analytics. The C-Code service provider pre-collects the signals of known contexts and tries to maximally discriminate two contexts. A simple user-input code (e.g., a one-digit letter) is assigned to each source in a context by the C-Code server, and is used to distinguish two sources in the same context. An information source is uniquely identified when both the context signal and the code match the record on the C-Code server. Our experiments demonstrate that C-Code provides an efficient way of accurately identifying information sources in various contexts, at the expense of simply inputting an easy-to-reproduce code.

Acknowledgements

This work was partially sponsored by National Natural Science Foundation of China under grant number 61201245 and 61272340, Specialized Research Fund for the Doctoral Program of Higher Education (SRFDP) under grant 20120001120128, and the Beijing Natural Science Foundation under grant 4143062.

References

[1] K. Antoni, B. Jakarta, R. Kosala, and K. Baru. Color Quick Response Code for Mobile Content Distribution. *ACM MoMM '09*, pp. 267–271, 2009.

[2] M. Azizyan, I. Constandache, and R. R. Choudhury. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. *Proc. MobiCom '09*, pp. 261–272, 2009.

[3] P. Bahl and V. N. Padmanabhan. RADAR: An In-Building RF-based User Location and Tracking System. *Proc. INFOCOM '00*, pp. 775–784, 2000.

[4] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha. FM-based Indoor Localization. *Proc. MobiSys '12*, pp. 169–182, 2012.

[5] Y. Chon, N. D. Lane, F. Li, H. Cha, and F. Zhao. Automatically Characterizing Places with Opportunistic CrowdSensing using Smartphones. *Proc. UbiComp '12*, pp. 481–490, 2012.

[6] J. Chung, M. Donahoe, C. Schmandt, I. Kim, P. Razavai, M. Wiseman. Indoor Location Sensing Using Geo-magnetism. *Proc. MobiSys '11*, pp. 141–154, 2011.

[7] S. Dhar, and U. Varshney. Challenges and Business Models for Mobile Location-based Services and Advertising. *ACM Communications*, 54(5):121–128, May 2011.

[8] T. Hao, R. Zhou, G. Xing. COBRA: Color Barcode Streaming for Smartphone Systems. *Proc. MobiSys '12*, pp. 85–98, 2012.

[9] P. Hu, L. Li, C. Peng, G. Shen, F. Zhao. Pharos: Enable Physical Analytics through Visible Light based Indoor Localization *Proc. HotNets-XII*, Nov. 2013.

[10] M. Kamvar, and S. Baluja A Large Scale Study of Wireless Search Behavior: Google Mobile Search. *Proc. SIGCHI '06*, pp. 701–709, 2006.

[11] Microsoft Research. Phytics: Physical Analytics. http://research.microsoft.com/en-us/projects/phytics/

[12] R. Nandakumar, S. Rallapalli, K. Chintalapudi, V. N. Padmanabhan, L. Qiu, A. Ganesan, S. Guha, D. Aggarwal, and A. Goenka. Physical Analytics: A New Frontier for (Indoor) Location Research. No. MSR-TR-2013-107, October 2013.

[13] S. Nath, F. X. Lin, L. Ravindranath, J. Padhye. SmartAds: Bringing Contextual Ads to Mobile Apps. *Proc. MobiSys '13*, pp. 111–124, 2013.

[14] V. Otsason, A. Varshavsky, A. LaMarca, and E. de Lara. Accurate GSM indoor localization, *Proc. UbiComp '05*, pp. 141–158, 2005.

[15] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: Zero-effort Crowdsourcing for Indoor Localization. *Proc. Mobicom '12*, pp. 293–304, 2012.

[16] S. Ramanathan. *Wireless Networks*, 5(2):81–94, March 1999.

[17] E. Rukzio, K. Leichtenstern, V. Callaghan, P. Holleis, A. Schmidt, J. Chin. An Experimental Comparison of Physical Mobile Interaction Techniques: Touching, Pointing and Scanning. *Proc. Ubicomp '06*, pp. 87–104, 2006.

[18] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, Y. Zhang. Walkie-Markie: Indoor Pathway Mapping Made Easy. *Proc. NSDI '13*, pp. 85–98, 2013.

[19] S. P. Tarzia, P. A. Dinda, R. P. Dick and G. Memik. Indoor Localization without Infrastructure using the Acoustic Background Spectrum. *Proc. MobiSys '11*, pp. 155–168, 2011.

[20] The global positioning system: a shared national asset: recommendations for technical improvements and enhancements. *National Academies Press*, pp. 16, chapter 1, 1995.

[21] Y. Tian, R. Gao, K. Bian, F. Ye, T. Wang, Y. Wang, X. LI. Towards Ubiquitous Indoor Localization Service Leveraging Environmental Physical Features. *Proc. IEEE INFOCOM* 2014.

[22] J. Xiong and K. Jamieson ArrayTrack: A Fine-Grained Indoor Location System. *Proc. NSDI '13*, pp. 71–84, 2013.

[23] C. Xu, B. Firner, R. S. Moore, Y. Zhang, W. Trappe, R. Howard, F. Zhang, N. An SCPL: indoor device-free multi-subject counting and localization using radio signal strength. In *Proc. IPSN '13*, pp. 79–90, 2013.

[24] S. Yoon, K. Lee, I. Rhee. FM-based indoor localization via automatic fingerprint DB construction and matching. In *Proc. MobiSys '13*, pp. 207–220, 2013.

[25] M. Youssef and A. Agrawala. The Horus Wlan Location Determination System. *Proc. MobiSys '05*, pp. 205–218, 2005.

[26] P. Zhou, Y. Zheng, Z. Li, M. Li and G. Shen. IODetector: A Generic Service for Indoor Outdoor Detection. *Proc. SenSys '12*, pp. 113–126, 2012.