Flash-Loc: Flashing Mobile Phones for Accurate Indoor Localization

Fan Yang[†], Qiang Zhai[†], Guoxing Chen[†], Adam C. Champion[†], Junda Zhu[‡], and Dong Xuan[†]
[†]Dept. of Computer Science & Engineering, The Ohio State University, USA
[‡]Faculty of Science and Technology, University of Macau, Macau, P.R. China
{yanfan,zhaiq,chenguo,champion,xuan}@cse.ohio-state.edu, jdzhu@umac.mo

***Abstract*—Accurate indoor localization is a key enabling technology for numerous applications such as indoor navigation, mobile social networking, and augmented reality. Despite major effort from the research community, state-of-the-art indoor localization performance remains unsatisfactory. Current approaches using radio frequency entail tedious site surveys and have limited accuracy. While vision-based localization techniques are promising, they struggle with human recognition and changing environments. This paper proposes *Flash-Loc*, an accurate indoor localization system leveraging flashes of light to localize people carrying mobile phones in areas with deployed surveillance cameras. A person's mobile phone emits a sequence of flashes that uniquely "represents" the person from the cameras' view. Flash-Loc develops three key mechanisms that distinguish people while avoiding long irritating flashes: adaptive-length flash coding, pulse width modulation based flash generation, and image subtraction based flash localization. Further, we design a system in which Flash-Loc cooperates with fingerprinting and dead reckoning for accurate human localization. We implement Flash-Loc on commercial off-the-shelf equipment. Our real-world experiments show Flash-Loc achieves accurate indoor localization by itself and in cooperation with other localization technology. In particular, Flash-Loc can localize a user 45 m away from the camera with sub-meter accuracy.**

I. Introduction

Accurate indoor localization enables a wide range of important applications including indoor navigation, mobile social networking, augmented reality, etc. [1], [2] Recently, significant effort has been devoted to this important enabling technology [3]–[10]. In addition, indoor localization is a promising global market for industry [11], which is estimated to exceed $4 billion by 2019 [12]. However, even with major effort from the research community and strong industrial demand, indoor localization remains very challenging and state-of-the-art performance remains unsatisfactory [7].

Existing work is based on different types of signals, including radio frequency (RF) [3], [4], vision [7]–[9], and others [5], [6]. Although RF-based indoor localization has been intensively studied for years, its performance is still unsatisfactory due to multipath, environmental change, human body interference, and other factors [1]. Besides, vision-based indoor localization approaches have been promising in recent years. Vision-based indoor localization has two steps: object recognition and positioning. However, both of these face significant challenges. In particular, object recognition is unreliable in changing environments and crowded areas due to visual occlusions. We discuss these challenges as well as other localization technologies in Section II.

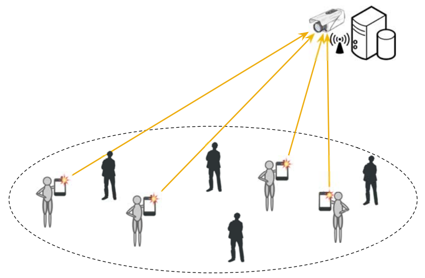


Fig. 1: Flash-Loc working scenario

In this paper, we present *Flash-Loc*, an indoor localization system that leverages light flashes to localize humans carrying mobile phones. Fig. 1 shows Flash-Loc's working scenario. In Flash-Loc, a human's mobile phone generates a sequence of flashes (discussed in Section III). Each sequence embeds carefully designed codes that uniquely represent a human. Flashes with abrupt brightness changes are clear visual indicators of users in the view of surveillance cameras in order to avoid unreliable human recognition. Thus, the cameras can accurately localize users by computing locations of visual flash points.

Since light flashes travel long distances before diminishing in intensity, our system can accurately localize a human with a mobile phone "flashlight" in a wide range of areas. Our real-world experiments show that our system can accurately localize a user 45 m away from commercial off-the-shelf cameras, which is far beyond most existing approaches like ultrasound-based localization.

In Flash-Loc, rapid localization is crucial in order to avoid irritating long-lasting flashes. The flash time of our system needs to be comparable to that in taking photos, which is around 2 s. However, several real-world challenges must be addressed to achieve this goal, including multiple users, false flashes, and missed or incorrect flash detections caused by environmental noise. To address them, we propose three key

The work was supported in part by the U.S. National Science Foundation (NSF) grants CNS-1065136, CNS-1218876, The Macau Science and Technology Development Fund under Grant FDCT023/2013/A1, and University of Macau Research Council under Multi Year Research Grant. Any opinions, findings, conclusions, and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

mechanisms: (1) adaptive-length flash coding, (2) pulse width modulation (PWM) based flash generation, and (3) image subtraction based flash localization. Our flash coding mechanism adaptively changes code lengths based on the number of concurrent users. To maximize our system's efficiency and scalability, especially with numerous users, flash codes are computed and assigned dynamically. We use PWM to modulate flash codes and carefully set the pulse parameters based on mobile phone LED flashes' and surveillance cameras' characteristics. We use image subtraction to achieve lightweight flash detection and localization. In addition, our flash detection process considers reflection effects and interference from other sources. Consequently, with the above three key mechanisms, our system can effectively address real-world challenges and accurately localize users in a timely manner.

Furthermore, to demonstrate Flash-Loc's feasibility working with other localization technologies, we design an accurate localization system via cooperation among Flash-Loc, RF-based fingerprinting, and dead reckoning. This system (described in Section IV) localizes human objects using RF fingerprinting as a default localization technology. Flash-Loc is sparsely deployed and infrequently used to avoid irritating flashes of light. However, Flash-Loc plays a significant role in this system. It serves as a "checkpoint" for the calibration and rectification of other localization techniques due to Flash-Loc's greater accuracy. In addition, we extend Flash-Loc's accurate localization result to a range rather than a point via dead reckoning.

We implement both Flash-Loc and our cooperative localization system with commercial off-the-shelf devices and conduct real-world experiments in typical indoor environments that consist of corridors, offices, and an open lobby with passersby. Our experimental results show that Flash-Loc achieves accurate localization in real-time under complicated scenarios with multiple users. Its average localization error is only around 0.5 m. It can localize single users in only 0.8 s and four users in under 2 s. Also, it works with other localization technologies and significantly improves overall localization accuracy by around 37%. Also, we have measured the power consumption of our Flash-Loc application on mobile phones.

The remainder of the paper is organized as follows. Section II reviews related work. Section III describes our Flash-Loc system design. Section IV discusses integration of Flash-Loc with other localization technologies. Section V elaborates our implementation and evaluation. Section VI concludes.

II. Related Work

This paper applies "smart" mobile phone and visual technologies for localization. There have been many publications on smart mobile phone applications and systems such as mobile health [13]–[15] and mobile social networking [16], [17]. Due to space limitations, this section only discusses related work on localization.

Radio-Frequency-Based (RF-Based) Localization: RF-based mechanisms have been intensively studied for indoor localization. Under widely deployed WiFi infrastructure, most existing approaches localize people via RF propagation models [18], [19] or RF fingerprinting [4], [20]. Model-based methods can only achieve room-level accuracy due to many factors such as human body interference, multipath, and background noise. Fingerprinting-based approaches achieve better accuracy by laborious crowdsourcing efforts to build electronic maps for fingerprinting. However, the accuracy of fingerprinting-based localization remains unsatisfactory as RF propagation is vulnerable to changing environments and the presence of people [1].

Vision-Based Localization: In vision-based indoor localization technology, localization has two steps: object recognition and positioning. Widely deployed indoor visual surveillance systems are used to recognize and position objects [21], [22]. Shen et al. [21] proposed an efficient background subtraction methods and developed a single object localization system in a 4 m × 4 m area with three cameras. Yu et al. [22] proposed a system for localizing multiple people by integrating several vision clues such as color, human detection, non-background information, and face recognition. Vision-based localization can achieve very high accuracy. However, unreliable human recognition can significantly affect its performance. To mitigate this problem, visible light is incorporated in localization [7], [23]. Epsilon [23] leverages LED beacons and applies trilateration for localization. Luxapose [7] uses modified LED luminaires as landmarks to localize mobile phones using angles of arrival. Most visible light based strategies require special or modified light sources or sensors. In contrast, Flash-Loc is implemented with commercial off-the-shelf devices without any hardware modification.

Localization Based on Sensor Fusion: Methods using sensor fusion are used for indoor localization. A combination of inertial sensors and cameras for self-localization is introduced in [24], [25]. Roetenberg et al. [26] proposed a localization system consisting of magnetic and inertial sensors. The system focuses on a person's motion rather than identifying him or her. EV-Loc [27] is a system that integrates electronic and visual signals for localization. EV-Loc requires one-to-one matching between electronic devices and visual objects, which is error-prone. EV-Loc requires accumulated E-V data to find the correct association, which is an inefficient approach. In contrast, Flash-Loc's approach is efficient as no data accumulation is required; hence, it can localize users within seconds.

III. Flash-Loc Design

A. Design Rationale

Flash-Loc is an indoor localization system that uses light flashes to localize people carrying mobile phones. Each person's mobile phone generates a sequence of flashes that embeds codes uniquely representing a person. Light travels in straight lines and resists multipath effects. Thus, surveillance cameras detect flashes' sudden brightness changes to localize people.

The key issue with Flash-Loc is the flash time. While we use short flashes in photography, long flashes of light are irritating.

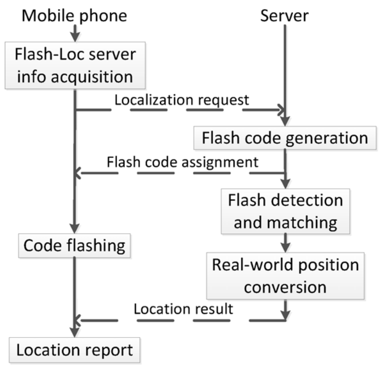


Fig. 2: Flash-Loc workflow

Our design goal is to minimize Flash-Loc's flash time so it is comparable to that in photography (about 2 s). There are several challenges to achieve this goal in real-world scenarios.

– First, multiple users may use Flash-Loc simultaneously. The flash time needs to be long enough to distinguish them.

– Second, visual signals are prone to environmental changes in brightness. Besides flashes from our system, some flashes serve other purposes (e.g., tourists taking photos). Glossy floors reflect fluorescent lights, creating *false flashes* that interfere with detection of Flash-Loc's *true flash*. Our system needs to recognize and handle these flashes.

– Third, missed or false detection of flashes can occur. Visual occlusion is a major cause of missed flash detections due to light's linear travel. Thus, our system cannot detect blocked flashes. Also, surveillance cameras' limited frame rates can cause our system to falsely detect flashing lights' on-off patterns, which can lead to incorrect user identification.

Flash-Loc addresses these challenges with three key mechanisms: adaptive-length flash coding, pulse width modulation (PWM) based flash generation, and image subtraction based flash localization. In adaptive-length flash coding, each user's flash has a distinct code, making multiple users distinguishable to cameras after the code is flashed only a few times. The code length changes adaptively according to the number of concurrent users. We use PWM to modulate flash codes, carefully setting pulse parameters that accord with mobile phone LED flashes' and surveillance cameras' characteristics. This creates distinct on-off flash patterns that are repeated quickly, distinguishing Flash-Loc's flashes. We use image subtraction in order to realize lightweight flash detection and localization. All these mechanisms reduce the probability of missed or false detections of flashes.

B. Workflow

In this subsection, we introduce Flash-Loc's system workflow, which Fig. 2 illustrates. There are two threads running on Flash-Loc: one on a mobile phone and the other on a server. The mobile phone thread is a mobile application communicating with the server via WiFi. The server manages cameras and serves users in the cameras' fields of view (FoVs). It knows each camera's position, orientation, and FoV.

To start Flash-Loc, the user needs to first acquire nearby camera information including whether the location is covered by a camera and the camera's facing direction. Visual markers can be deployed as camera indicators similar to common street signs saying "Smile! You are on camera." We can make cameras visually prominent such that users can directly find cameras. Another method to acquire camera information is to use other "ready-made" localization approaches to estimate users' locations; users can then retrieve this information from central servers. Section IV elaborates this approach.

After acquiring camera information, the mobile phone sends out a request to the server to initialize the localization process. The server registers the user in a user-in-service table and assigns a flash code to the user. (Flash codes are temporary user IDs.) The mobile phone then flashes repeatedly according to the flash code. Since each user is assigned a locally unique code, the server is able to distinguish concurrent flashes from different positions. After detecting a sequence of flashes at one position, the server decodes the flashes and matches the code against all flash codes in the user-in-service table. The position is converted from image pixels to real-world coordinates and reported to the user with the "best match" flash code.

Three main functional modules are involved in the entire procedure as mentioned in Section III-A: flash coding, flash generation, and flash localization. The details of these three main functional modules are presented in Sections III-C, III-D, and III-E, respectively.

C. Flash Coding

This subsection elaborates Flash-Loc's adaptive-length flash coding. To minimize flash time, our coding scheme uses variable length and "circularly equivalent" codes.

1) Variable-Length Codes: Due to Flash-Loc's ad hoc usage, the number of concurrent users may vary greatly. Thus, fixed-length codes may be a poor choice. For example, if a system needs to handle at most 256 users, the lengths of fixed-length codes must be 8 or more. However, we may have only two users for half the time, in which case codes of length 1, i.e., 0 or 1, suffice for localization causing the least irritation.

Another approach is the dynamic use of different fixed-length codes according to the number of current users. The problem is that individual users may need to keep changing their codes when the number of other users varies. This may cause confusion and even localization failure. Enabling users to "keep" their codes before finishing use of this service is very desirable.

Hence, we design a variable-length code that assigns shorter codes to the first few users and longer codes to the following users as the number of concurrent users increases. The basic idea is as follows: as the number of users increases and codes of a certain length are depleted, the code length is doubled and new codes can be generated while the original users' codes automatically transition to codes of the new (doubled) length by repetition. To achieve such an automatic length transition property, mobile devices are required to repeat their codes until localization results are received. Algorithm 1 generates such a code space C_{assign} that assigns codes to users.

Example: Codes 0 and 1 are assigned to the first two users. When the third user arrives, the code length is doubled to 2 and a new code 01 is assigned to the third user while the previous codes 0 and 1 transition to 00 and 11, respectively. The first two users still think their codes are 0 and 1, but the server uses 00 and 11, respectively, to localize them.

Algorithm 1 Generate variable length codes

```
1: $C_{assign} := \{0, 1\}$
2: $C_{current} := C_{assign}$
3: $L = 1$
4: for $i := 1$ to $n$ do
5:     Replace every code $c$ in $C_{current}$ with $c||c$
6:     $L = L \times 2$
7:     $C_{complete} := \{0, 1\}^L$
8:     $C_{new} := C_{complete} - C_{current}$
9:     $C_{assign} := C_{assign} + C_{new}$
10:    $C_{current} := C_{current} + C_{new}$
11: end for
```

No waiting and recycling: Using these variable length codes, the server can serve any incoming user immediately. (Later, we will discuss delaying some requests in order to potentially reduce the overall flash time.) When users finish localization, their codes are released. Any incoming user is assigned the shortest available code and the server uses the maximum length of all assigned codes for decoding.

2) Circularly Equivalent (CE) Codes: Due to background noise and missed detections, errors may occur when decoding. Using "repetitive flash" mode, the server collects multiple repetitions before confirming the code. This is a tradeoff between accuracy and irritation.

However, synchronization may be a major issue for such a repetition-based scheme. If one bit is missing and the server fails to notice this, all of the following repetitions will be decoded to some "rotation" of the original code. For example, when a user's code is 01, the repetitions are 01010101. If the first bit is missing, the received bits are 1010101, which decodes to 10.

As the incorrectly decoded code is a rotation of the original code, we propose using *circularly equivalent* (CE) codes (Definition 1) to address the synchronization problem. Intuitively, we regard one code and its CE codes as a single code that identifies a user. Thus, when a code is assigned to one user, its CE codes (or rotations) cannot be assigned to other users. For example, if 0001 is assigned, 0010, 0100, and 1000 cannot be assigned.

Definition 1: **Circularly Equivalent (CE) Codes**: Two codes $m, m' \in \{0, 1\}^L$ are *circularly equivalent* (CE) if and only if there exists an integer j s.t. $m_i = m'_{(i+j) \bmod L}, i = 0, 1, \ldots, L-1$, denoted as $m \equiv m'$ (where $m = (m_0, m_1, \ldots, m_{L-1})$ and $m' = (m'_0, m'_1, \ldots, m'_{L-1})$).

We modify Algorithm 1 to generate variable-length CE codes in Algorithm 2.

For each code c of length L, there are at most L CE codes. Thus, the cardinality of the code space of length L is at least $2^L/L$. Roughly speaking, codes of length L incur a cost of $\log L$ bits to handle the synchronization problem.

Algorithm 2 Generate Circularly Equivalent (CE) Codes

```
1: $C^c_{assign} := \{0, 1\}$
2: $C^c_{current} := C^c_{assign}$
3: $L = 1$
4: for $i := 1$ to $n$ do
5:     Replace every code $c$ in $C^c_{current}$ with $c||c$
6:     $L = L \times 2$
7:     $C^c_{complete} := \{0, 1\}^L$
8:     for $c$ in $C^c_{current}$ do
9:         delete $c$ and its CE codes from $C^c_{complete}$
10:    end for
11:    while $C^c_{complete}$ is not empty do
12:        choose code $c$ in $C^c_{complete}$, add $c$ to $C^c_{assign}$ and $C^c_{current}$
13:        delete $c$ and its CE codes from $C^c_{complete}$
14:    end while
15: end for
```

Example: When $L = 4, 6$ codes are available: $\{0, 1, 01, 0001, 0011, 0111\}$.

Next, we analyze the code error rate. Let p_e denote the probability of a single bit error. The error rate of one code of length L with 1 repetition is: $p_c^{(1)} = 1 - (1 - p_e)^L$. Let k denote the number of repetitions. When $k > 1$, repetitions are required for decoding. Successful decoding occurs when the kL bits are k repetitions of a certain code: either all k corresponding repeated bits of every bit of the code are correct or none of them is correct. Thus, the probability of successful decoding is: $P^{(k)}_{decoded} = (p_e^k + (1 - p_e)^k)^L$.

The probability of correct decoding is $P^{(k)}_{correct} = (1-p_e)^{kL}$. It follows that the error rate with k repetitions is: $p_c^{(k)} = (P^{(k)}_{decoded} - P^{(k)}_{correct})/P^{(k)}_{decoded} = 1-(1-p_e^k/(p_e^k+(1-p_e)^k)^L$.

This scheme cannot correct or detect any errors because the minimum distance (or Hamming distance) of any two codes is 1. Code spaces should be designed with large minimum Hamming distances. Due to CE codes, we use *circular Hamming distance* to measure the distance of codes.

Definition 2: **Circular Hamming Distance**: The *circular Hamming distance* d_{CH} of two codes m_1 and m_2 is defined as the minimum Hamming distance d_H of m_1 and m_2 CE codes: $d_{CH}(m_1, m_2) = \min_{m \in \{m || m \equiv m_2\}} d_H(m_1, m)$.

As a simple improvement, we only use codes with even Hamming distances. A single bit error results in odd Hamming distance that we can detect. Another improvement is to limit the total number of concurrent users and delay any incoming user before the number of concurrent users drops below the limit. The intuition is that given a bit error rate, longer codes are error-prone, thus causing more disturbance.

D. Flash Generation

In this subsection, we detail Flash-Loc's flash generation (via PWM). Flash generation designs optimal flash patterns to

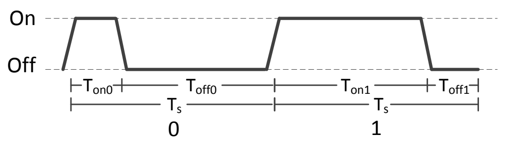


Fig. 3: Flash symbolization

carry coding information that represents each user's identity. We first discuss the optimal flash mode selection, then decide the key parameters that represent each bit of information.

Flash Mode Selection: Flash codes are carried by physical flashes. The most straightforward approach (using different flash frequencies) does not work well for multiple users due to hardware limitations (i.e., mobile phones' limited real-time flashlight control capability with low camera frame rates). Another approach is to use a flash "on" to represent 1 and a flash "off" to represent 0. However, coding information is undetectable if the on-off state never changes. For example, the code 0000 will cause a long flash "silence," which makes the code very difficult to detect. Thus, we use two adjacent flash on-off switches as a single symbol of information since cameras can easily detect these flash switches. Furthermore, we use a fixed length of time for each bit. The difference between 0 and 1 is the proportion of "on" time for this fixed bit period (i.e., the duty cycle). We call this scheme *flash pulse width modulation* (FPWM). Compared with variable flash length, where the difference between 0 and 1 is the bit period, FPWM can help estimate the number of missed detected bits between two adjacent detected bits caused by occasional occlusion or flash orientation deviation.

Flash Parameter Determination: In FPWM, the duty cycle conveys the 0-1 symbol. Fig. 3 illustrates flash symbolization in Flash-Loc. A fixed length on-off switch is designed as a 0-1 symbol. The length of one symbol is T_s. For 0, the on and off times are T_{on0} and T_{off0}, respectively, where $T_{on0} + T_{off0} = T_s$. For 1, the on and off times are T_{on1} and T_{off1}, respectively, where $T_{on1} + T_{off1} = T_s$. Every symbol starts with a rising edge and ends before another rising edge. The difference between T_{on0} and T_{on1} is used to distinguish 0 and 1. The fixed symbol period can help to count missed detections. If the time t between two detected rising edges exceeds $2T_s$, there are approximately $t/T_s - 1$ missed symbols.

Precise control of the flash switching time is critical for FPWM. To reduce the total localization time, the flash frequency should be as high as possible in order to achieve high bit rate, which is challenging for time control. However, common mobile phones cannot precisely control switching LED flashes on and off. We conduct several tests of flash control on Android mobile phones. The results show that a single "switch-on" or "switch-off" operation costs 2–60 ms. We set $T_s = 200$ ms and $T_{on} = T_{off} = 100$ ms. Fig. 4 shows the actual time between two switch-on operations for 200 tests. All these times exceed 200 ms with a 26 ms average delay and a 113 ms maximum delay. This control delay may increase if the CPU is heavily loaded. Clearly, there is considerable

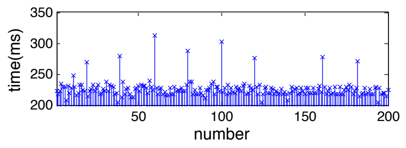


Fig. 4: Flash control uncertainty

uncertainty in controlling flash generation time.

In flash detection, each video frame can be treated as a sample and the camera's frame rate is the sample rate. With larger frame rates, we can reduce the period T_s of each symbol so that less flash time is needed to convey a flash code. Let T_f denote the time between consecutive video frames. To distinguish 0 and 1 as defined in Fig. 3, we should set $T_{on0} = T_{off1} = T_f$ and $T_{off0} = T_{on1} = 2T_f$ to minimize the time span of one symbol. However, mobile phones cannot precisely control timings of flash "switch-on" and "switch-off" and the server may not receive each visual frame uniformly due to inaccurate camera timing control and unpredictable network transmission delay. Both of these factors cause decoding errors. Enlarging the difference between T_{on0} and T_{on1} eliminates such errors but increases flash time for localization. Thus, we set $T_{on0} = T_{off1} = T_f$ and $T_{off0} = T_{on1} = 3T_f$ so that the difference between the "on times" is $2T_f$, which tolerates frequency shifts on both the mobile phone side and the camera side.

E. Flash Localization

In this subsection, we detail flash localization (via image subtraction). Flash localization has two parts: flash detection and flash positioning.

Flash Detection: Flash detection is the first step in flash localization. Here, we detect abrupt brightness changes in consecutive video frames. Spots with regular brightness changes are recorded as flashes. As Fig. 3 shows, the flash between two consecutive switch-ons is extracted as one symbol. To do so, all RGB video images are first converted to grayscale. We subtract two consecutive grayscale images and perform contour detection after thresholding to find spots of sharp brightness variation. In addition to true flashes, brightness changes caused by object movements and occasional reflections can be detected.

Environmental noise from other light sources and multipath reflection are two main causes of false detection. Flash noise is unlikely to follow the pattern shown in Fig. 3 with designated "on" and "off" times. Hence, flash symbolization can help eliminate flash noise. Flash reflections caused by walls and floors generate false flashes that follow true flashes' pattern exactly. As a result, we can detect multiple legal flashes; however, only one of them is the correct one. We use shape detection and location filtering to handle this problem. The shape of true flashes generated by the mobile phone LED light is nearly circular. Reflected flashes on floors and walls tend to have irregular shapes. Moreover, most reflected flashes tend to be localized outside the camera's coverage area. For example, a flash position caused by wall reflection will be

Distance	10 m	15 m	20 m	25 m	30 m
Angle	$\sim 60°$	$\sim 54°$	$\sim 51°$	$\sim 46°$	$\sim 43°$

TABLE I: Detectable flash distances and angles

on the opposite side of the wall. This also helps recognize reflected flashes.

Flash distance and orientation to the camera are the main factors that determine whether a flash is detectable. Long distances and large orientation angles make the flash weaker and the effective flash area on the camera smaller. As a result, the camera may recognize the flash as noise. We test the flash detection algorithm's tolerance to distance and orientation angle. Test users hold a flashing mobile phone and rotate their facing angles (relative to the camera) from $90°$ to $0°$, recording critical angles where the camera detects the flash. Table I shows the test results. Our flash detection algorithm still works well when the mobile phone is 30 m away at a $\sim 43°$ angle from the camera, which loosens constraints on users' relative positions.

Flash Positioning: After flash detection, we localize the phone by finding the flash's pixel location in video frames as well as the projective transformation between real-world and image pixel locations. First, we find intrinsic and extrinsic parameters according to the pin-hole camera model [28]. With these parameters, we can map the image plane to a real-world 2D plane. We only need to perform this camera calibration once for our system.

In our system, depending on camera coverage, one or more cameras may detect the flash simultaneously. We design different positioning methods to fully use information from all detected cameras. If a single camera detects the flash, we can find the mobile phone's real-world location if its height is known. In Flash-Loc, users hold the mobile phone in front of their faces to flash. If the user's height is given, we can roughly estimate the mobile phone's height. Height estimation may be inaccurate, which introduces localization error. If the height estimation error is Δh and the camera's elevation angle is θ, the location error will be $\Delta d = \Delta h \cot \theta$. In typical practical settings, if $\Delta h = 15$ cm and $\theta = 30°$, then location error is 26 cm. This means that a reasonable human height estimation introduces acceptable error. If two cameras detect the flash, we can determine the mobile phone's true 3D position with stereo vision technology.

IV. Integration with Other Localization Technologies

In this section, we present Flash-Loc's significant role in cooperation with existing localization systems. As mentioned above, Flash-Loc can localize a person in the covered area accurately and efficiently. On the other hand, Flash-Loc's limitations (cost of system establishment and irritation arising from frequent use) may hinder its deployment in large areas. However, even with its sparse deployment and infrequent use, its accuracy and robustness afford an opportunity for cooperation with other existing localization systems that can significantly improve overall localization accuracy.

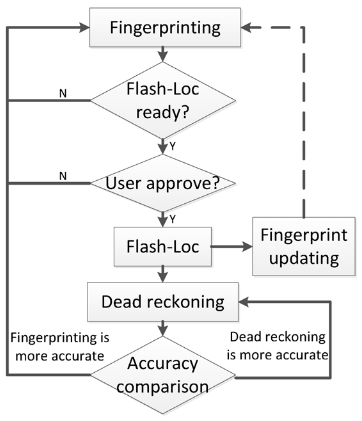


Fig. 5: Augmented Flash-Loc workflow

To demonstrate Flash-Loc's feasibility for this purpose, we design an accurate reliable localization system via cooperation among Flash-Loc, fingerprinting, and dead reckoning. In this system, we localize users mainly via fingerprinting because it is resilient to accumulated error drift over time. Initially, our system only has a coarse fingerprinting map for rough localization to avoid exhaustive war-driving. To assist fingerprinting while avoiding irritating flashes of light, Flash-Loc is sparsely deployed and infrequently used. Due to Flash-Loc's accuracy, it serves as a "checkpoint" for fingerprinting calibration and rectification. A location's electronic fingerprints "checked" by Flash-Loc updates the fingerprinting map. Multiple users' updates gradually improve the overall localization accuracy. Furthermore, we extend Flash-Loc's impact via dead reckoning, which transforms "checkpoints" to "check zones," accelerating improvement in overall localization accuracy. However, such a system requires laborious establishment and suffers visual occlusions. The localization system proposed in this section uses effective strategies for high localization accuracy, lightweight system establishment, and minimal user irritation.

Fig. 5 illustrates the workflow of our cooperative localization system. Our system has surveillance cameras that capture flash signals for localization. These cameras are sparsely deployed and they do not cover the entire area. Also, our system has several access points (APs) for communications and fingerprinting signal sources. Each user of our system has a mobile phone with a WiFi antenna and inertial sensors for fingerprinting and dead reckoning, respectively.

Our system's fingerprinting-based localization approach derives from Horus [4]. To localize a fingerprint, our system chooses the k fingerprints in the map with the largest posterior probability and computes the centroid of these as the localization result. Error estimation for our localization approach follows that of k-nearest neighbors (the average distance from the centroid to the k nearest locations).

Our dead reckoning localization approach localizes users by counting their steps and accumulating each user's step displacement, which can be estimated via two consecutive Flash-Loc localizations. We use a low-pass filter, sliding window, and adaptive thresholding to process mobile phones' noisy accelerometer data. In practical experiments, we find this approach is very accurate and the step count error is below 2%. Our system also uses mobile phones' gyroscopes to detect users' turns and calculate their directions.

The key step in the workflow shown in Fig. 5 is accuracy comparison. We need to estimate fingerprinting and dead reckoning errors and determine whether the accuracy is acceptable. We treat the average distance from the localization result to the k-nearest neighbors' locations as fingerprinting error. We consider two types of errors for error estimation of our dead reckoning localization approach: displacement error and direction error. We assume the displacement error of each step follows an i.i.d. distribution; we compute its average and deviation from two adjacent Flash-Loc localizations. The direction error is an integration of each sensing deviation over time. Each gyroscope's sensing deviation is a calibrated parameter that can be acquired from its specifications. For example, the MEMS gyroscope sensor in a Nexus S mobile phone is an STMicrocelectronics L3G4200D whose rate noise density is $50\,\mathrm{dps}/\sqrt{\mathrm{Hz}}$ with 50 Hz bandwidth [29]. Thus, the error of our dead reckoning localization approach can be modeled as the combination of displacement error under a Gaussian distribution and increasing direction error over time.

V. Implementation and Evaluation

In this section, we discuss the implementation of Flash-Loc and its integration with other localization technologies. Next, we present our experimental evaluation.

A. Implementation

The implementation of our Flash-Loc system has two main components: the user-side mobile phone flash control program and the server-side flash localization service.

The user-side mobile phone program has three functional modules: (1) *Flash module:* Switch the flash on or off following a specific flash code until the location is obtained; (2) *Fingerprinting module:* Scan WiFi APs and collect RSSI information, then upload WiFi fingerprints to the server in order to find locations; (3) *Dead reckoning module:* Mobile localization with inertial sensing data. These functional modules are coordinated via the workflow shown in Fig. 5. We implement these functions on Nexus S mobile phones running Android 2.3+.

On the server side, we use commodity cameras (D-Link DCS-930L) to monitor the area of interest. We set the cameras' frame rates to 15 fps, which is consistent with those of real-world surveillance cameras. We send video frames to the server via Ethernet. We perform flash detection, decoding, and localization on a laptop with a dual-core Intel Core i5 CPU and 4 GB RAM. We implement the server software in Python using OpenCV 2.4.9 on Ubuntu Linux 12.04.

B. Evaluation

We evaluate the performance of both Flash-Loc and the cooperative localization system via real-world experiments. For Flash-Loc, we evaluate single-user localization accuracy and flash time at different distances. We also measure our system's flash time when there are multiple users. For the cooperative localization system, we evaluate its accuracy and compare it with that of pure fingerprinting and dead reckoning. The experimental setting is shown in Fig. 6; it is a 50 m $\times$ 10 m lobby on the first floor of a building on a university campus. Occasional passersby are treated as interference. Three cameras are placed in a line at one end of the lobby on the third floor, which is 6 m high. (The cameras are shown as a triangle in Fig. 6.) The distance between cameras is 2.5 m. The cameras at the two ends form a two-camera checkpoint covering a space 10–35 m away. The camera in the middle individually covers a space up to 50 m away. We adopt Horus [4] and implement a fingerprinting localization system. Fingerprinting signatures are collected in the lobby with a data density of 3 m per data point. The fingerprint database has RSSI values from 20 APs.

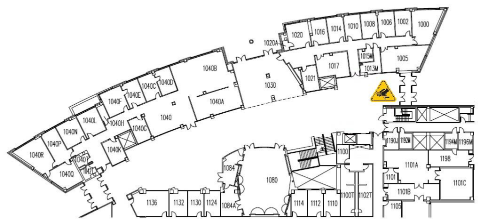
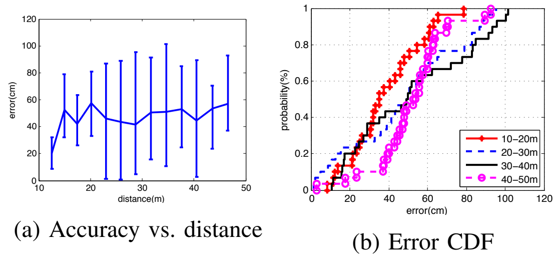


Fig. 6: Experiment environment



(a) Accuracy vs. distance

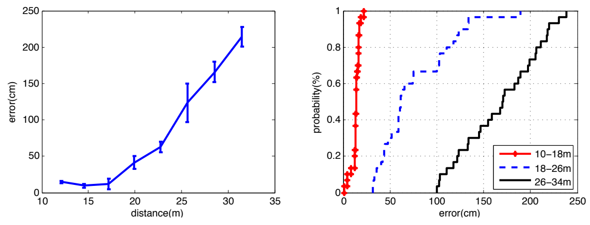
(b) Error CDF

Fig. 7: One camera localization accuracy

1) Evaluation of Flash-Loc: We evaluate Flash-Loc in both single-user and multi-user cases.

Single-User Flash-Loc Accuracy at Different Distances: We first evaluate Flash-Loc's localization accuracy in the basic single-user case. One user carries a mobile phone and requests the Flash-Loc service at different distances from a camera covering the area.

If only one camera sees the user, then the user's location is calculated with an estimated human body height that the user provides. The user's distance from the camera ranges from 12–46 m. The average, minimum, and maximum localization errors at different distances are shown in Fig. 7a. The smallest error occurs at the nearest distance; on average, it is only 19.9 cm. The average error at farther distances is larger, but it never exceeds 60 cm. As discussed in Section III-E, the camera calibration error and human body height estimation error are the root causes. Fig. 7b shows that CDFs of errors at different ranges overlap. Uncertainty regarding the difference between body height estimation and the mobile phone's true height introduces significant variation in the results. However, all errors at 90% for different ranges are less than 90 cm.



(a) Accuracy vs. distance (b) Error CDF

Fig. 8: Two camera localization accuracy

If more than one camera covers the user simultaneously, height estimation is unnecessary. Fig. 8a shows the localization error with two cameras and 12–32 m user distances from the cameras. The best localization accuracy (10 cm) occurs when the user distance is under 18 m. The error rises when the distance exceeds 20 m, reaching a maximum of 2 m when the user distance is 32 m. As shown in Fig. 8b, the error at 90% is 20 cm for distances below 18 m and increases to 2.2 m for distances above 30 m. This is due to two cameras' accumulated calibration errors, which greatly increases the localization error compared to the one-camera case, especially at larger distances.

The above results show that our Flash-Loc camera can cover a large range of areas efficiently.

Single-User Flash-Loc Time: The flash time needed to obtain location results is a particular metric for Flash-Loc evaluation. This directly relates to irritation posed by light flashes. We test the flash time at varying distances (12–32 m) from the camera in both one-camera and two-camera cases. In the one-camera case, the user may not flash directly toward the camera, but may face slightly left or right of it. In the experiment, flash orientation angles with respect to the camera vary among $0°$, $15°$, and $30°$. Table I shows the results. In the two-camera case, errors at different angles are evaluated together. Fig. 9 shows the flash time results. Generally, flash time increases with distance. Flash time is ~ 800 ms at 12 m and below 1500 ms at 32 m in the two-camera case and in the one-camera case for small angles. When the angle is $30°$ in the one-camera case, the flash time increases, but it remains within $\sim$2 s in the worst case. Both large distances and angles reduce effective detection areas for flashes, increasing the probability of missed detection, which prolongs the flash time.

Multi-User Flash-Loc Time: Compared with single users, multiple concurrent users need longer flash codes to distinguish them from each other as discussed in Section III-C. To test the impact, we have two, three, and four users request the Flash-Loc service simultaneously at distances 10 m, 20 m, and 30 m from the cameras as well as in two separate groups (one half 15 m from the cameras, the other half 25 m from them). In the experiment, the distance between users in one group is 1 m. Fig. 10 shows the flash time results for multiple users. We see that flash time increases with distance and achieves its minimum at 10 m from the camera. For the three-user and four-user cases, the flash time increases more with distance than in the two-user case. This happens for the following reason: when the third user requests the Flash-Loc service, the flash code length increases from 2 to 4 bits and more time is needed to identify the users. The gap between the two-user and the three-user cases is about the time of two flash bits (600 ms). As distance increases, the gap also increases because longer flash codes cause more decoding errors. This experiment demonstrates that users in a group localize themselves quickly and Flash-Loc works well with nearby user interference.

2) Evaluation of Integrated Localization System: In the following experiments, users randomly walk in the lobby as shown in Fig. 6. We record and compare localization results of our cooperative system, pure fingerprinting (Horus [4]).

The overall localization error distribution shown in Fig. 11 has a median error of 1.7 m in our system, which is about 1 m better than pure fingerprinting results. The accuracy improvement is mainly caused by the "checkpoint" and "check zone" effect proposed in Section IV. The Flash-Loc checkpoints and calibrated dead reckoning check zones improve the overall localization accuracy. In fact, the RSSI signatures at the checkpoints and check zones are uploaded to the server to update the whole fingerprinting database, which improves long-term fingerprinting accuracy.

We collect a typical sample of users walking in the lobby via pure fingerprinting, dead reckoning, and cooperative localization. Fig. 12 shows localization accuracy vs. time. Here, the user requests the Flash-Loc service at times 1 s, 29 s, and 42 s (denoted by the red dashed circles in Fig. 12). In our system, the error sharply drops below 50 cm whenever Flash-Loc completes, then follows the error of dead reckoning. After some time, the confidence of dead reckoning becomes lower than that of fingerprinting and fingerprinting error dominates the overall localization error. Pure dead reckoning error decreases from 16 s to 30 s. This is because the user is at the same position at times 1 s and 30 s and motion sensor error diminishes when the user returns to the start point. This result shows that the localization error is very small after Flash-Loc. Dead reckoning achieves high accuracy in a short time after calibration by Flash-Loc. Pervasive fingerprinting performs localization after the dead reckoning error accumulates to some extent. This demonstrates that the three localization components in our system are highly complementary.

3) Evaluation of Power Consumption: To test power consumption of the Flash-Loc application running on mobile phones, we monitor power states on Nexus S phones for three hours under different scenarios. (1) *Standby mode:* the Nexus S phone remains in standby mode; (2) *Flash-Loc mode:* the Flash-Loc application runs on the Nexus S phone collecting WiFi RSSI signatures, performing dead reckoning, and flashing upon request every minute; and (3) *Constant flash mode:* the Nexus S phone turns its flash on and off every second. The phone has a 1500 mAh Li-ion battery. Fig. 13 shows power consumption with time. With ongoing accelerometer and RSSI measurements, the Flash-Loc application lasts about 15 h before the battery is exhausted. In addition, the flash clearly consumes a large portion of battery power when it keeps flashing. Flash-Loc's flash time needs to be reduced in

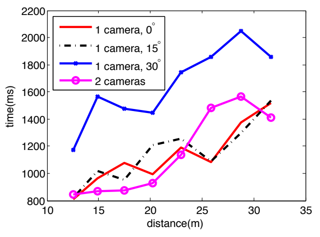


Fig. 9: Flash time vs. distance and angle

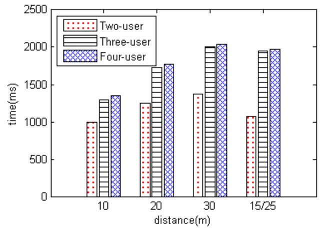


Fig. 10: Multiuser flash time vs. distance

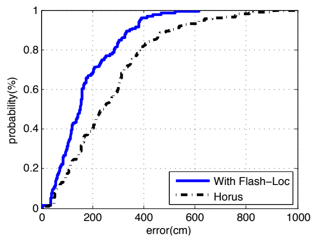


Fig. 11: Overall error CDF

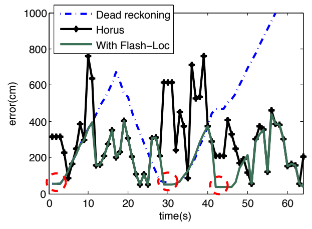


Fig. 12: Accuracy improves with time

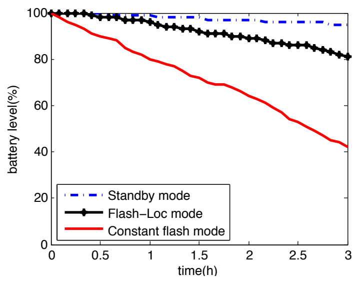


Fig. 13: Power consumption

order to minimize power consumption.

VI. Conclusions

This paper proposed *Flash-Loc*, an accurate indoor localization system leveraging flashes of light that localized people carrying mobile phones in areas with deployed surveillance cameras. Each person's mobile phone emitted a sequence of flashes that uniquely represented the person from the cameras' point of view. Flash-Loc developed three key mechanisms that distinguished people while minimizing flash time: adaptive-length flash coding, pulse width modulation (PWM) based flash generation, and image subtraction based flash localization. Further, we designed a system using Flash-Loc, fingerprinting, and dead reckoning in cooperation for timely localization of people. We implemented Flash-Loc on commercial off-the-shelf equipment. Our real-world experiments showed Flash-Loc achieved accurate indoor localization alone and in cooperation with other localization technologies. In future work, we will further study Flash-Loc with other localization technologies such as acoustic localization and depth cameras.

Acknowledgments

We thank Prof. Yuan F. Zheng and Yiran Xuan for their assistance with this paper.

References

[1] L. Li, G. Shen, C. Zhao, T. Moscibroda, J.-H. Lin, and F. Zhao, "Experiencing and Handling the Diversity in Data Density and Environmental Locality in an Indoor Positioning Service," in *MobiCom*, 2014.

[2] S. Kumar, S. Gil, D. Katabi, and D. Rus, "Accurate Indoor Localization with Zero Start-up Cost," in *MobiCom*, 2014.

[3] P. Bahl and V. N. Padmanabhan, "RADAR: An In-Building RF-based User Location and Tracking system," in *INFOCOM*, 2000.

[4] M. Youssef and A. Agrawala, "The Horus WLAN Location Determination System," in *MobiSys*, 2005.

[5] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," in *MobiCom*, 2000.

[6] K. Liu, X. Liu, and X. Li, "Guoguo: Enabling fine-grained indoor localization via smartphone," in *MobiSys*, 2013.

[7] Y.-S. Kuo, P. Pannuto, K.-J. Hsiao, and P. Dutta, "Luxapose: Indoor positioning with mobile phones and visible light," in *MobiCom*, 2014.

[8] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[9] Y. Tian, R. Gao, K. Bian, F. Ye, T. Wang, Y. Wang, and X. Li, "Towards ubiquitous indoor localization service leveraging environmental physical features," in *INFOCOM*, 2014.

[10] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to war-drive: Unsupervised indoor localization," in *MobiSys*, 2012.

[11] K. Dennehy, "2014: Big Move to Retail Indoor Location Market," Dec. 2014, http://gpsworld.com/2014-big-move-to-retail-indoor-location-market/.

[12] MarketsandMarkets, "Indoor Location Market by Solution, by Application, by Service, by Vertical, & by Region - Global Forecast Up to 2019," Nov. 2014, http://www.marketsandmarkets.com/.

[13] J. Dai, J. Teng, X. Bai, Z. Shen, and D. Xuan, "Mobile Phone Based Drunk Driving Detection," in *PervasiveHealth*, 2010.

[14] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, "PerFallD: A Pervasive Fall Detection System Using Mobile Phones," in *PerCom Workshops*, 2010.

[15] S. Kumar, W. Nilsen, M. Pavel, and M. Srivastava, "Mobile health: Revolutionizing healthcare through transdisciplinary research," *Computer*, vol. 46, no. 1, pp. 28–35, 2013.

[16] J. Teng, B. Zhang, X. Li, X. Bai, and D. Xuan, "E-Shadow: Lubricating Social Interaction using Mobile Phones," in *ICDCS*, 2011.

[17] N. Jabeur, S. Zeadally, and B. Sayed, "Mobile social networking applications," *Commun. ACM*, vol. 56, no. 3, pp. 71–79, Mar. 2013.

[18] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, and M. Corner, "Virtual Compass: Relative Positioning to Sense Mobile Social Interactions," in *Proc. Pervasive*, 2010.

[19] N. Bulusu, J. Heidemann, and D. Estrin, "Gps-less low-cost outdoor localization for very small devices," *IEEE Personal Commun.*, vol. 7, no. 5, pp. 28 –34, oct 2000.

[20] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: Wireless indoor localization with little human intervention," in *MobiCom*, 2012.

[21] Y. Shen, W. Hu, J. Liu, M. Yang, B. Wei, and C. T. Chou, "Efficient Background Subtraction for Real-Time Tracking in Embedded Camera Networks," in *SenSys*, 2012.

[22] S.-I. Yu, Y. Yang, and A. Hauptmann, "Harry Potter's Marauder's Map: Localizing and Tracking Multiple Persons-of-Interest by Nonnegative Discretization," in *CVPR*, 2013.

[23] L. Li, P. Hu, C. Peng, G. Shen, and F. Zhao, "Epsilon: A visible light based positioning system," in *NSDI*, 2014, pp. 331–343.

[24] E. Foxlin, L. Naimark *et al.*, "Vis-tracker: A wearable vision-inertial self-tracker." *VR*, vol. 3, p. 199, 2003.

[25] T. Teixeira, D. Jung, and A. Savvides, "Tasking networked cctv cameras and mobile phones to identify and localize multiple people," in *UbiComp*, 2010.

[26] D. Roetenberg, P. J. Slycke, and P. H. Veltink, "Ambulatory position and orientation tracking fusing magnetic and inertial sensing," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 5, pp. 883–890, 2007.

[27] B. Zhang, J. Teng, J. Zhu, X. Li, D. Xuan, and Y. F. Zheng, "EV-Loc: Integrating Electronic and Visual Signals for Accurate Localization," in *MobiHoc*, 2012.

[28] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

[29] STMicroelectronics, "L3G4200DMEMS motion sensor: three-axis digital output gyroscope," http://www.st.com/web/catalog/sense_power /FM89/SC1288/PF250373.