# Pair-Navi: Peer-to-Peer Indoor Navigation with Mobile Visual SLAM

Erqun Dong*§, Jingao Xu*§, Chenshu Wu†, Yunhao Liu*‡, Zheng Yang*
*School of Software and BNRist, Tsinghua University
†Department of Electrical & Computer Engineering, University of Maryland, College Park
‡Department of Computer Science and Engineering, Michigan State University
§Co-primary author
{doneq13, xujingao13, wucs32, yunhaoliu, hmilyyz}@gmail.com

***Abstract*—Existing indoor navigation solutions usually require pre-deployed comprehensive location services with precise indoor maps and, more importantly, all rely on dedicatedly installed or existed infrastructure. In this paper, we present *Pair-Navi*, an infrastructure-free indoor navigation system that circumvents all these requirements by reusing a previous traveler's (*i.e.* leader) trace experience to navigate future users (*i.e.* followers) in a Peer-to-Peer (P2P) mode. Our system leverages the advances of visual SLAM on commercial smartphones. Visual SLAM systems, however, are vulnerable to environmental dynamics in the precision and robustness and involve intensive computation that prohibits real-time applications. To combat environmental changes, we propose to cull non-rigid contexts and keep only the static and rigid contents in use. To enable real-time navigation on mobiles, we decouple and reorganize the highly coupled SLAM modules for leaders and followers. We implement *Pair-Navi* on commodity smartphones and validate its performance in three diverse buildings. Our results show that *Pair-Navi* achieves an immediate navigation success rate of 98.6%, which maintains as 83.4% even after two weeks since the leaders' traces were collected, outperforming the state-of-the-art solutions by $> 50\%$. Being truly infrastructure-free, *Pair-Navi* sheds lights on practical indoor navigations for mobile users.**

## I. Introduction

During the past decades, technologies using Wi-Fi [1] [2] [3], RFID [4] [5], sound [6] and visible lights [7] *etc.*, have been proposed to shape a range of location-based services. Therein, indoor navigation with a smartphone acts as a killer application. All conventional navigation techniques, however, require particular infrastructure, either pre-existing or dedicatedly installed, to be appropriately set up in advance in the area-of-interests. Recently, an alternative Peer-to-Peer (P2P) navigation is proposed to circumvent the pre-installation of indoor localization services [8]. In this mode, a previous traveler, named *leader*, records the trace information (*e.g.*, turnings and certain ambient properties) and shares it through the Internet to a later *follower*, who needs to travel to the same destination. A typical example would be a self-deployed navigation service to direct a customer to a shop, which enables a shop owner to record such trace information from the entrance of a large mall to his/her own shop and offer them to potential visitors as guidance, without resorting to any pre-deployed location systems provided by third parties.

Several pioneer works have demonstrated such a leader-follower mode for P2P navigation [8]–[10]. These works mainly leverage ambient Wi-Fi signals, in addition to inertial sensor measurements and/or images captured by smartphone

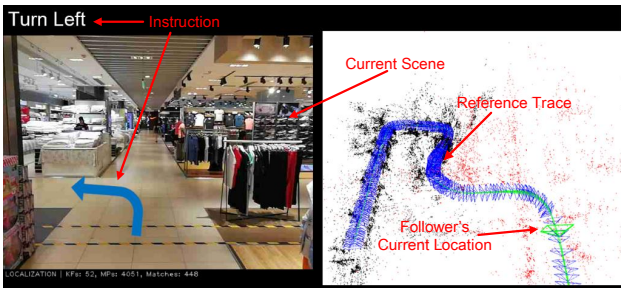



Fig. 1. Follower's navigation interface of *Pair-Navi*

[11], as trace properties to synchronize leaders' and followers' traces. Although these works are inspiring, the pre-installation of Wi-Fi and the error introduced by the dynamically changing nature of Wi-Fi signal and the inertial sensors make them less-than-ideal for practical usage.

Recently, two arising trends may overcome the above limitations and underpin a practical solution to indoor navigation. First, simultaneous localization and mapping (SLAM) technology has been rapidly developed. For example, visual SLAM has been enabled with a single camera [12]–[14], making it feasible on commodity smartphones that usually have only one camera on the back side. Second, vision capability has become as a standard and continues growing more powerful on mobile devices, allowing advanced vision tasks on mobiles.

In this work, we investigate visual SLAM with the power of mobile vision and present *Pair-Navi*, a P2P indoor navigation system that requires no pre-existing or dedicatedly installed infrastructure, pre-deployed localization service, or indoor digital maps. Visual SLAM utilizes one or more cameras to explore an unknown environment by continuously locating the cameara itself in the environment and meanwhile constructing a map of the environment [14]. Our approach is built upon monocular visual SLAM with a single camera commonly equipped on commodity smartphones. A leader of *Pair-Navi* simply walks through a path recording a video clip along the route and shares the trajectory video via a cloud server for potential upcoming followers. *Pair-Navi* consumes the video for SLAM and constructs the trajectory that the leader has traveled. When a follower appears, he/she will be provided with a leader's trajectory as reference. On the follower side, *Pair-Navi* also captures real-time video frames and precisely locates the follower's relative location to the reference trajectory, accordingly navigating the follower by

timely promoting walking hints. As shown in Figure 1, both the current scene and the reference trace will be displayed to the follower. In addition, *Pair-Navi* handles navigation deviation, which is necessary but neglected by existing P2P navigation approaches. Without the need to instrument the building-of-interests, *Pair-Navi* works in any scenes as long as a camera-equipped smartphone is available.

However, translating visual SLAM into a robust navigation system entails various challenges: 1) *Environmental Non-Rigidity*. Most popular SLAM solutions assume rigid and static indoor environments, where the surrounding scenes are not supposed to change both when the SLAM is running and after the map is constructed [13]–[15]. However, the real world is time-varying due to considerable dynamics, *e.g.*, pedestrians, furniture changes, advertising screens, the inherent object deformation, and lighting condition variations, rendering the constructed trajectories inaccurate and difficult to follow. As shown in Figure 2, such environmental non-rigidity will cause errors in video frame matching and thus significantly degrading visual SLAM. Although some SLAM approaches attempt to reason about minimal non-rigidity with restrictive applicability [14], [16], it still remains challenging to employ visual SLAM for mobile indoor navigation in vibrant scenarios full of dynamics, such as busy shopping malls and large airports, *etc.* 2) *Real-time*. Visual SLAM technologies typically require intense computation for several core tasks including visual odometry and optimization, making them difficult to run in real-time on commodity mobiles. A practical navigation application, however, should locate the user precisely, render the navigation path, and provide user-friendly instructions, all in real-time. Applying visual SLAM to real-time mobile navigation is a non-trivial task that calls for significant efforts in system design and implementation.

To combat environmental non-rigidity, we propose to extract and subtract the dynamic foregrounds, *e.g.*, pedestrians and other changing contents involved in the trajectory video and keep only the rigid parts for SLAM. The key observations are two-fold: 1) Video frames of typical indoor environments usually provide abundant features for SLAM, allowing room to sift out non-rigid contexts while keeping as good or even better performance since the remained features are mainly from those rigid and reliable objects. 2) Recent progresses in computer vision, especially with the application of deep learning, make it feasible for efficient and effective detection and segmentation of non-rigid dynamic contexts inside a video [17]–[19]. Based on these two insights, we employ Mask Region-based CNN (Mask R-CNN) [19] to identify non-rigid objects and cull them from the video feature set used for SLAM. By doing so, *Pair-Navi* eliminates the impacts of the environmental dynamics, thus retaining robust trajectories for leaders as well as precise locations for followers.

To enable real-time navigation on smartphones, *Pair-Navi* decouples the originally coupled SLAM modules and re-assembles merely the necessary modules. For a follower, rather than employing a complete SLAM system, we only conduct relocalization to synchronize his/her relative walking progress to a leader's trajectory. Furthermore, we employ a synchronization strategy for the basic visual navigation module and non-rigid context culling module. By doing so, *Pair-Navi* achieves real-time navigation on a mobile. In contrast, the latest works [20]–[22] that attempted to incorporate semantics for robust SLAM fail to operate in real-time.

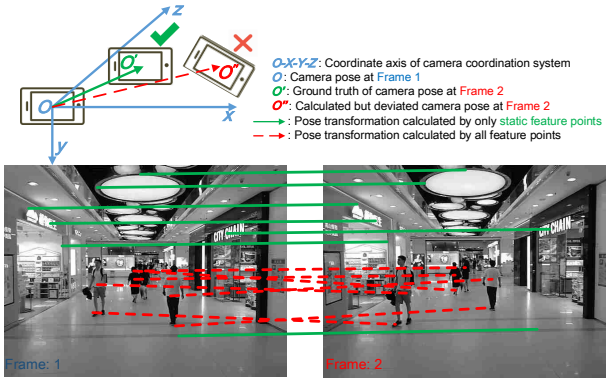



Fig. 2. Illustration of feature point matching of two consecutive frames. Feature points from rigid contexts are matched correctly (solid green lines), while most of feature points from non-rigid contexts are mismatched (dashed red lines), resulting in errors in camera pose calculation.

We implement our system on the Robot Operating System (ROS) platform [23] on the server and on ROS-Android [24] on the phone side. Comprehensive experiments are carried out in three buildings with various conditions over two weeks. The results demonstrate that *Pair-Navi* achieves a remarkable navigation success rate of 98.6%. Even after two weeks since the construction of the leaders' trajectory, the rate maintains 83.4%, outperforming the state-of-the-art *Travi-Navi* [10] by 50.9% and *FollowMe* [9] by 80.4%. Being truly infrastructure-free, *Pair-Navi* takes an important step towards practical indoor navigation for mobile users.

In summary, the core contributions are as follows.

- We present the first vision-based P2P navigation system, which neither requires to instrument a building nor relies on pre-deployed localization service with indoor maps.
- We employ non-rigid context culling by using Mask-RCNN to overcome indoor environmental dynamics for visual SLAM, which significantly improves the robustness and precision of navigation.
- We implement a complete real-time system on commodity smartphones and extensively evaluate the performance. The results show *Pair-Navi* achieves delightful results and outperforms all existing solutions.

The rest of paper is organized as follows. We present an overview in Section II and introduce visual navigation in non-rigid environment in Section III. Real-time design and implementation is provided in Section IV, followed by experiments in Section V. We review related works in Section VI and conclude in Section VII.

## II. Overview

### A. Peer-to-peer Navigation

Different from conventional navigationsystems that rely on pre-deployed localization services, *Pair-Navi* works in an easy-to-deploy P2P navigation mode. P2P navigation also circumvents the need of indoor digital maps, which are sometimes difficult to obtain and process. There are two key roles in a P2P

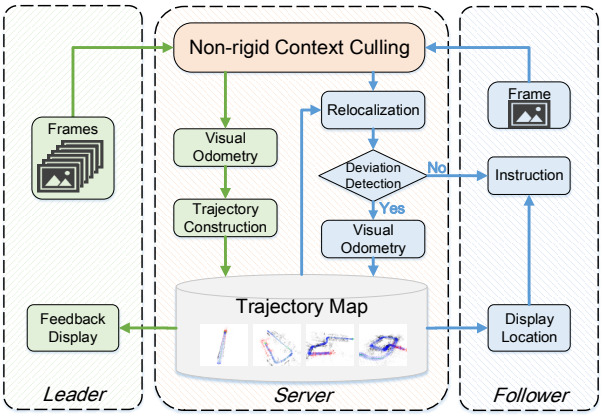



Fig. 3. *Pair-Navi* architecture

navigation system, i.e., a leader and a follower. The basic idea is to reuse the experience from earlier travelers who become leaders. Anyone walked through a path can serve as a leader for that particular path by contributing corresponding trace information, i.e., certain trace data (*e.g.*, Wi-Fi signal series, goemagnetic series and IMU sensor measurements [8]–[10], or video clips in our case) together with the automatically extracted walking hints (*e.g.*, heading, turning, climbing, *etc.*). The trace information is later requested by and sent to a follower for his/her reference. The navigation for the follower is then achieved by synchronizing his/her relative location to a leader's reference trajectory. Note that a user can participate as either a leader or a follower, depending on specific scenarios.

P2P navigation is in particular useful as a fast- and easy-to-deploy service for ordinary users who demand to provide small-scale navigation. For example, a shop owner can provide a self-owned navigation service to guide potential customers to his/her own shop, and a conference organizer can direct attendees to the conference location with little efforts. Among many other similar scenarios, P2P navigation, which is self-deployable and almost zero-effort, acts as a promising alternative to traditional centralized localization and navigation systems.

### B. System Overview

*Pair-Navi* enables this kind of leader-follower navigation by leveraging mobile vision capabilities. The system architecture is illustrated in Figure 3. For both leaders and followers, they walk naturally in the course and hold their smartphones to shoot videos along the trace. Every video frame captured by his/her smartphone camera is sent to a cloud server via network for further processing. Although we leverage SLAM technology, our system does not involve all modules for leader and follower. For a leader, we feed the video clips into two SLAM modules, i.e., *Visual Odometry* and *Trajectory Construction*, to simultaneously form a trajectory (a sequence of 3D camera poses) and construct a map (3D map points and key frames). When the trace is completed, the trajectory and map data, in addition to leader-labeled starting and destination places, will be stored on the server.

In case a follower arrives, he/she first chooses the destination and will be provided with a reference trace leading to the same destination, contributed by some leader. When the follower walks, our system will immediately locate his/her relative location to the reference trace by *relocalization* based on the currently captured video frame. If relocalization succeeds, the follower will be relatively located and accordingly instructed with timely walking hints that come with the leader's reference trace. However, if relocalization fails, which indicates the follower may deviate from the given path in the following steps, a *deviation detection* module will be triggered to launch auxiliary visual odometry to track the follower's camera poses independently. The results will also be fed back and displayed on the follower's phone, together with the reference trace, so that the follower can get himself back on the correct path for further navigation.

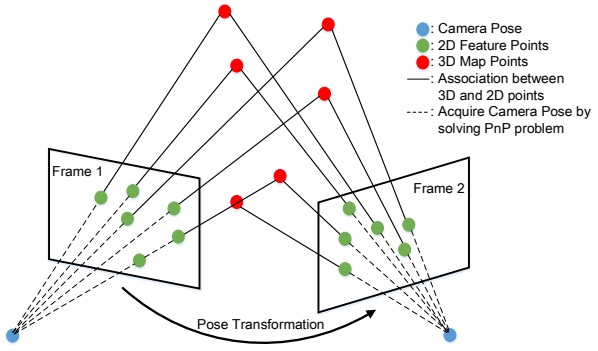



Fig. 4. Illustration of Visual Odometry

A key and unique component in *Pair-Navi* is the *Non-Rigid Context Culling* (NRCC), which aims to extract and subtract dynamic contents in the video clips to combat time-varying environments. To ensure precise reference trajectory generation and robust follower localization, NRCC is applied to both leader's and follower's videos.

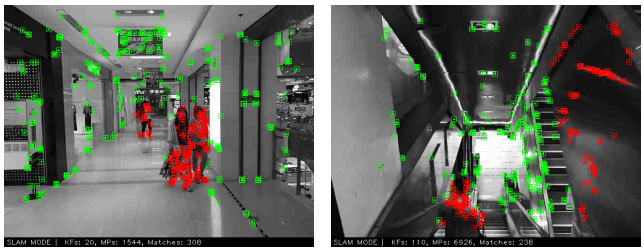
## III. Visual Navigation in Non-rigid Environment

In this section, we describe how *Pair-Navi* utilizes visual SLAM for navigation and how it addresses non-rigidity in the environment.

### A. Visual SLAM for Navigation

In *Pair-Navi*, we use monocular visual SLAM that works with a single camera since most smartphones have only one camera on the back side. We decouple the tightly coupled modules of a monocular visual SLAM system, and reorganize the required modules into leader and follower applications to simplify for real-time meanwhile ensure accuracy. In particular, we introduce two key modules, visual odometry and relocalization, as follow.

**Visual Odometry.** To begin with, our system takes an initialization step using epipolar geometry [25] [14] to locate the camera in an initial map with 3D map points as landmarks of the environment. The visual odometry (VO) module takes charge of the system thence, to continuously track the camera pose from consecutive video frames. Specifically, when a new video frame arrives, its 2D feature points are extracted and associated to already-created 3D map points by feature matching. We choose ORB feature point [26] for this purpose



(a) (b)

Fig. 5. Examples of non-rigid contexts: feature points from non-rigid contexts, pedestrians in (a) and mirror reflections in (b), are recognized and marked as red, while the remaining feature points from rigid and static objects are green.

since it simultaneously yields sufficient matching accuracy and efficient computation.

Figure 4 shows example of the extracted feature points associated to 3D map points. From the association of 2D feature points and 3D map points, we can acquire the camera pose of this frame by solving a so-called Perspective-n-Point (PnP) problem [27], which determines the position and orientation of a camera from a set of correspondences between 3D points and 2D pixel points. As shown in Figure 4, given the camera poses of two frames, new 3D map points can be generated by calculating the 3D coordinates via triangulation among the two frames [25]. Repeatedly, as more new frames come, a trajectory of the camera poses and a map of the 3D landmarks and corresponding keyframes[1] are built incrementally. In *Pair-Navi*, VO is the core module for a leader to construct a trajectory map.

**Relocalization.** In order to reuse a previously built trajectory map, relocalizaiton module comes in handy, which is the central component in the follower program. It compares a video frame with the keyframes in the map, and finds out the most similar keyframe based on feature point matching. This step is also called visual place recognition, of which the state-of-the-art is Bags of Visual Words [28]. After the most similar keyframe is found, the feature points in the current follower frame is associated to the feature points in the selected keyframe. On this basis, a PnP problem [27] is solved in the same manner as VO module to get the camera pose, thus relocalizing the camera in the map. In *Pair-Navi*, the follower program mainly employs relocalization to achieve efficient relative localization (Section IV).

Note that classical monocular visual SLAM still involves complicated steps like loop closing detection, global optimization, *etc.* [14]. In our system, however, we merely apply the necessary modules for leaders and followers respectively to avoid intensive computation, as detailed in Section IV.

### *B. Non-Rigid Context Culling (NRCC)*

*1) Limitations of Visual SLAM in Non-Rigid Environments:* While visual SLAM technology underpins a promising solution to infrastructure-free navigation, it is vulnerable to non-rigid indoor environments with significant dynamic changes over time. Specifically, the limitations are two-fold:

**Low-Precision Trajectory.** As is described in the above section, to calculate the camera poses, we need to match feature points first. Therefore, correct feature point matches influences the accuracy of the constructed trajectory. In presence of erroneous matches, the generated trajectory will deviate from the ground truth. Figure 2 illustrates an example of matching two consecutive video frames from a typical shopping mall. As seen, feature points extracted from those dynamic contexts (*e.g.*, pedestrians) will lead to considerable erroneous matching, as indicated by red lines in Figure 2. As a consequence, if we calculate camera poses from the whole set of feature point matches without screening, the constructed trajectory will deviate from the truth, depraving further navigation. To obtain precise trajectory, we need to intelligently recognize the non-rigid contexts and sift out their corresponding feature points.

**Vulnerable Relocalization.** Apart from degrading trajectory precision, non-rigid contexts further harm relocalization robustness. In practice, the environment observed by a leader and later a follower may change significantly, leading to feature point mismatches and thus large relocalization errors or even relocalization failures. In one situation, if there are only a small number of matching outliers, the feature points may be matched to wrong 3D map points, resulting in errors in camera pose computation. In another situation, if a large portion of feature points fail to match, a wrong keyframe will be chosen and relocalization fails.

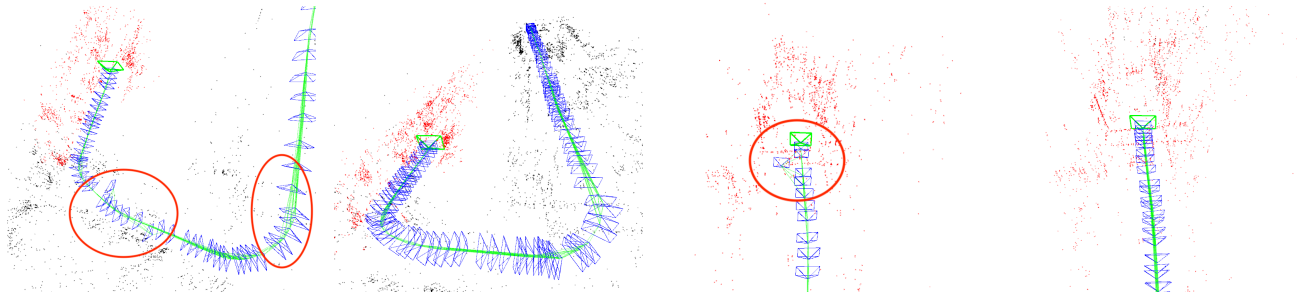
Therefore, to ensure accurate trajectory construction for leader as well as successful relocalization for follower navigation, we propose the non-rigid context culling (NRCC) module that takes out features points from non-rigid contexts and only exploits the remaining reliable feature points, mainly from rigid and static areas, for frame matching. Our key observation is that there are abundant feature points for frame matching, allowing room to cull part of them without degrading visual SLAM performance.

*2) Non-Rigid Context Culling via Mask R-CNN:* In *Pair-Navi*, we adapt Mask R-CNN [19] for NRCC. Mask R-CNN is a recent framework for instance segmentation. It aims to separate different instances in an image via a segmentation mask for each instance. We use the Mask R-CNN network pre-trained on COCO dataset [29], and select the object categories that are suitable for indoor scenario.

Since we aim at distinguishing rigid and non-rigid context, we divide all the object categories into two sets: rigid context objects and non-rigid context objects as shown in Table I. If an object belongs to the rigid context set, it means the location, pose and shape of the object will not change, and whenever the leader or follower come to the same place, they will observe the object in the same situation. On the contrary, an object is dynamic if it belongs to the non-rigid context set. What is worth mentioning is the classification of the two sets is flexible. Some objects (*e.g.* vase, potted plant) can belong to either the rigid context object set or the other, depending on the time interval between the leader's trajectory being constructed and follower's navigation. For example, for vases, if the time interval is shorter than 7 days, they will be regarded as rigid context objects. Yet if the interval is longer than 7 days, they will be treated as non-rigid context.

When the server receives a video frame from the camera,

[1]Keyframes are a subset of all frames to eliminate redundancy.



(a) (b) (c) (d)

Fig. 6. The effect of NRCC on trajectory construction: A complex trajectory computed (a) without NRCC, in which the camera poses of some frames in the red circle are deviated from the original path, and (b) with NRCC, in which the camera poses are correct and smooth. Similarly, (c) and (d) show a simple trajectory without and with NRCC, respectively.

we extract its feature points and use Mask R-CNN framework to detect the non-rigid contexts, then we filter out the feature points that lie in the masks of dynamic instances. As shown in Figure 5, the feature points filtered out, which are marked by red color, are from mainly from people, and the map is generated only using the feature points belonging to static environment, which are marked by green. Moreover, we also use the method proposed in YOLO [30] to detect mirrors and smooth surfaces in video frames (as shown in Figure 5b). We believe the illumination change in places like academic buildings may be drastic, rendering the feature points lying in mirrors volatile. Therefore they are also culled to increase system robustness. After this preprocessing of the video frame, although the user may be facing a non-rigid environment, the feature points lying in the masks of dynamic instances will not be involved in trajectory construction (for leaders) or relocalization (for followers). As shown in Figure 6, removing non-rigid contexts helps improve precision and avoid potential relocalization failures.

## IV. Real-time Navigation

In this section, we present the design of *Pair-Navi* to enable real-time navigation.

### A. Follower Real-time Navigation

In the follower navigation program, the smartphone runs as a ROS node, captures video frames, and sends them to the server. The server, as another ROS node, runs relocalization, renders the visualization of follower's camera pose in the leader's trajectory map, and calculates the navigation instruction.

The navigation instruction is calculated as this: the server averages the next 10 keyframe poses in the leader's trajectory as temporary navigation destination, and, for example, if the temporary navigation destination is on the left of the follower's camera, the server will give an instruction of "turn left". On receiving all the returned messages, the follower can see the navigation instruction, together with an visualization of the current camera pose and leader's trajectory map.

If the relocalization observes inadequate quantity of 3D map points, the auxiliary VO is launched from this frame, and will take place of the relocalization to show camera pose if it truely fails. At this time, the follower can still see the camera pose in the leader's trajectory and spare little effort to return to the course. Once the relocalization is successful again, the auxiliary VO is shut down, and the follower goes back to the normal case, in which only relocalization is executed.

Our system has the following three designs that ensure it can run in real-time (by "real-time", we mean at least 10 fps, the typical value of vision persistence):

**Relocalization.** We decouple a typical visual SLAM system, adaptively combining the relocalization module and VO module into our follower navigation program. In this design, we avoid the heavy overhead to acquire the follower's current position with a full visual SLAM system since the beginning of follower's navigation. Instead, we get the utmost of the leaders' efforts, only running relocalization for follower, which is more efficient than a full visual SLAM system yet produces as accurate results as it. As will be demonstrated in Section V, our design considerably reduces running latency compared to a conventional SLAM system.

**Mask Synchronization Strategy.** Non-rigid context culling is for both leader and follower's video frames, but in different ways. For leaders, we generate one mask for each frame, because the leader's map construction can be done offline. Yet for followers, the calculation of Mask R-CNN is relatively slow, at an average frame rate of about 5 fps. So we take a trade off strategy to synchronize Mask R-CNN with visual SLAM. We aggregate every 2 frames (corresponding to 0.2s due to 10fps), and generate one mask for them both.

The rationale behind is that a user will move for only about 20cm during the 0.2s time-window, assuming a typical walking speed of 1m/s. Therefore the potential scene changes in two consecutive frames will not be too significant, which may slightly affect trajectory construction accuracy for leaders, but do not necessarily influence relocalization for followers. In summary, to obtain a highly precise and reliable trajectory for followers to use, we need to cull non-rigid contexts for each frame; yet to save the the computational resources for real-time follower navigation, we can confidently reuse the masks for several consecutive frames.

**The fringe benefit of NRCC.** Additionally, NRCC brings some fringe benifit to the real-time performance. Since the number of feature points are reduced, further computation, including feature point matching, calculating bags of visual words, PnP, *etc.*, is simplified by the proportion of the rigid

TABLE I
OBJECT CLASSIFICATION FOR INDOOR SCENARIO

| Non-Rigid Context objects | Rigid Context objects |
|---|---|
| person, hair drier, toothbrush, cat, keyboard, phone, bottle, apple, cup backpack, umbrella, handbag, tie, suitcase, dog, frisbee, book, clock, skis snowboard, sports ball, kite, bat, glove, skatebord, surfboard, mouse, remote glass, fork, knife, spoon, bowl, banana, tennis racket, scissors, teddy bear sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake, chair, laptop | couch, potted plant*, dining table tv, microwave, oven, toaster refrigerator, vase*, bed, toilet, sink |

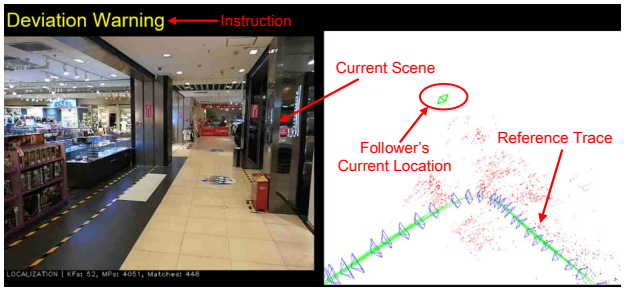



Fig. 7. Illustration of auxiliary VO when deviation happens

feature points in all the feature points.

### B. Follower Deviation Handling

Existing P2P navigation systems such as *Travi-Navi* and *FollowMe* have no function of deviation handling, and thus if follower deviation happens, the tracking will lose and the navigation will fail, with no recovery approach. In contrast, we add an auxiliary VO module to track the deviated trajectory, which acts as a fail-safe measure to guarantee the success of navigation.

In our scenario, deviation happens when the follower walks off the instructed course, or even when camera pose slightly deviates. In these cases, the follower's relocalization module observes inadequate 3D map points in a frame, which is when the system launches the auxiliary VO. The auxiliary VO, extracting new feature points and generating new 3D map points, runs in parallel with the relocalization to retain continuous tracking and show the camera pose in the trajectory map, as shown in Figure 7. With the camera pose shown in the trajectory map, the follower can easily go back on the right track. Once the relocalization observes a healthy quantity of 3D map points, the auxiliary VO will shutdown and handover the navigation to the relocalization module.

Note that to further keep our system running in real-time with this auxiliary VO, we strictly select a feature point number threshold for triggering it, so that redundant calculation is eliminated. In this way, we make the system tolerate navigation deviations in the wild while still maintaining real-time performance.

### C. Implementation

We implement our system on ROS Kinetic platform. The user program is developed on ROS-Android platform. The server's SLAM program is developed on *ORB-SLAM* [14] on ROS, and we develop the system visualization upon the visualization of *ORB-SLAM*, with some modifications via OpenCV. We resize all the frames to $640 \times 480$, which is relatively high-resolution and still not too large to prolong the process time of each frame. The phone-server communication resorts to ROS topic publicaction and subscription, which guarantees all the imformation is integral because ROS data transfer is based on TCP protocal.

We applied our Mask R-CNN models with the ResNet-FPN-50 backbone and the network parameters are pre-trained on COCO image Dateset. The Mask R-CNN code is implemented in python-3.6.5 with pytorch-0.4.0.

## V. EXPERIMENTS AND EVALUATION

### A. Experiment Settings and Methodology

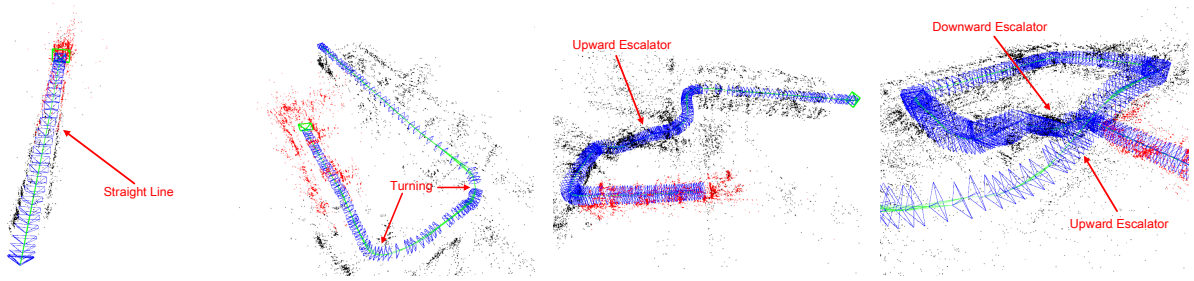
**Experiment Venues.** We conducted extensive experiments in an office building, a gymnasium and the 1st-4th floor of a shopping mall, with area sizes of about $400m^2$, $1{,}000m^2$ and $6{,}000m^2$, respectively. The three testing environments have diverse conditions. In particular, the crowded shopping mall is the most dynamic. The office building has the most drastic illumination oscillation during a day. The gymnasium has the medium crowdedness among the three environments.

**Data Collection.** Overall, we design 21 navigation paths, including 6 short paths ($\leq 100m$), 7 medium paths ($100m - 200m$) and 8 long paths ($\geq 200m$), covering all the main pathways of the testing areas. Figure 8 shows four trajectories of different lengths constructed from the three areas. The reference trajectories for these paths are constructed by 3 different leaders. The total length of leaders' reference trajectories is about 3.1km with $33,452$ video frames, among which $6,781$ keyframes are selected, and the followers' total walking distance is around 15km.

**Devices.** We tested *Pair-Navi* on a variety of Android mobile devices, including Huawei P10, Nexus 6p, Nexus 7, and Lenovo Phab2 pro. Since the main device discrepancies are camera intrinsics (*i.e.* focal length, lens center and distortion), we calibrate camera intrinsics of the smartphones and accordingly rectify video frames. The server, Lenovo IdeaPad-Y700 with i7-6700HQ CPU of 2.6GHz main frequency and 8G RAM, runs the Ubuntu 16.0.4 operating system and ROS Kinetic. For Mask R-CNN, the GPU we used is TITAN V with cuda version 9.1.85 and cudnn-7.05.

**Users.** We recruited 4 volunteer followers to walk along different routes naturally as they usually do. The follower behaviors are diverse in camera holding gestures and heights. For example, one user prefers to hold the camera with two hands, while others tend to use their right hands. Thus the cameras suffer from various extents of shake when the users are walking, which may cause different feature point matches. User study is conducted on three particular days: the same day as the trajectories were constructed, one week later and two weeks later.

**Comparison.** To evaluate the performance of *Pair-Navi*, we implemented *Travi-Navi* [10] and *FollowMe* [9], two start-of-the-art P2P navigation systems for comparison.




(a) (b) (c) (d)

Fig. 8. Four typical real trajectories in our experiment: (a) a straight line; (b) a U-turn; (c) a complex trajectory going up an escalator; (d) a complex trajectory going up one escalator and then down another

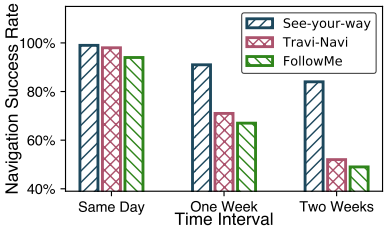



Fig. 9. Different methods

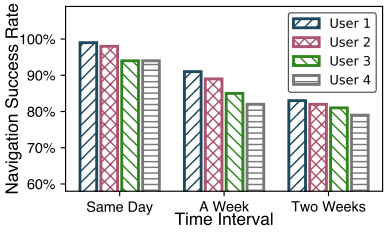



Fig. 10. Different users and time interval

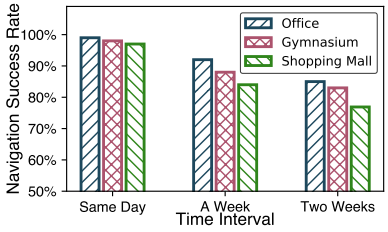



Fig. 11. Different areas and time interval

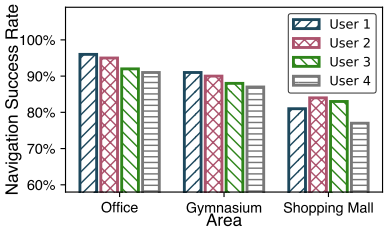



Fig. 12. Different users and areas

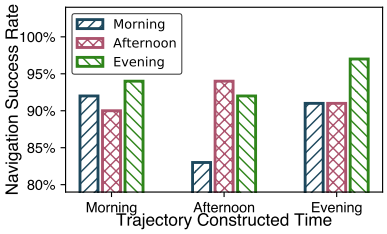



Fig. 13. Different time during a day

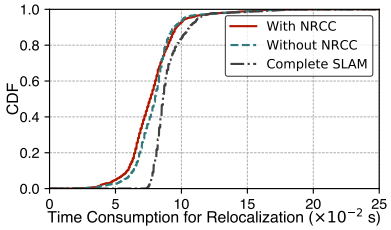



Fig. 14. System latency

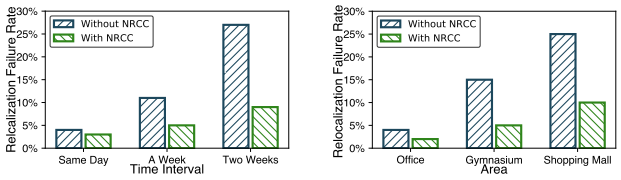
**Evaluation Metrics.** Similar to some existing works like *Travi-Navi* and *FollowMe*, we set checkpoints at turns, escalators and some landmarks on each trajectory. In total, we set 274 checkpoints for the 21 navigation paths. The followers were not informed of navigation routes, the final destination, or the checkpoint locations. *Navigation success rate* is defined as the rate of successful arrival at each checkpoint in *Travi-Navi* and *FollowMe*. Thanks to the employment of deviation handling, this rate is always 100% in *Pair-Navi*, which means the followers arrived at the destinations successfully in all cases. So instead, we use a more strict definition of navigation success rate as $1 - p$ (where $p$ is the rate of auxiliary VO launches) for *Pair-Navi*; while keep the original defination of navigation success rate unchanged for *Travi-Navi* and *FollowMe*.

### B. Overall Performance

*1) Performance Comparison:* The performances of *Pair-Navi* as well as the two state-of-the-art approaches to compare are depicted in Figure 9. We find that *Pair-Navi* achieves the best performance among all three of them, no matter how long the time interval is. The average navigation success rates by *Pair-Navi* in the same day of the trajectory's construction, after one week and after two weeks are 98.6%, 93.2% and 83.4%, respectively. Compared with the immediate performance (tested in the same day), the navigation success rates after two weeks decline 14.1%, 49.3% and 59.3% in *Pair-Navi*, *Travi-Navi* and *FollowMe*. In contrast to *Travi-Navi* and *FollowMe*, *Pair-Navi* attains high navigation accuracy after two-week interval, and outperforms *Travi-Navi* by 50.9% and *FollowMe* by 80.4%. The performance gains come from not only the advantages of our vision-based design over previously radio- and sensor-based methods, but also the robustness introduced by NRCC.

*2) Performance under Different Conditions:* We invite four volunteers to examine the robustness and practicability of *Pair-Navi* in different areas and at different times. As shown in Figure 10, *Pair-Navi* achieves an average navigation success rate of more than 85% for each user and the decrease of success rate for each one is less than 15% after two weeks. Furtherore, Figure 11 shows that *Pair-Navi* yields similar performance regardless of the different crowdedness levels and illumination conditions at different areas. *Pair-Navi* achieves consistently delightful success rate of more than 90%, 85% and 80% in the office building, gymnasium and large shopping mall under time interval within two weeks.

To further demonstrate the applicability of *Pair-Navi*, we note that the trajectories in the office building are mainly constructed by User-2 and User-3, and in the gymnasium and shopping mall by User-1 and User-2. The heights of the four users are different, and camera holding gestures are variant. Figure 12 reflects the robustness of *Pair-Navi* for different leaders and followers. Navigation success rates for all users in different areas are more the 80% and the success rate gaps




(a) Different time interval (b) Different areas

Fig. 15. Impact of NRCC on relocalization failure rate.

between different users are within 5%.

*3) Impact of NRCC:* To demonstrate the effect of non-rigid context culling, we compare the relocalization failure rate with and without non-rigid context culling. Specifically, we save all the video frames captured by follower's camera and record whether each frame can be relocalized (matched to a keyframe in the trajectory map with enough feature point matches). The experiment is conducted on all 21 navigation trajectories and under different time intervals.

As shown in Figure 15a, the relocalization failure rate increases from 4% to 27% without NRCC in the same day, one week later and two weeks later, while keeps low at 3%, 5% and 9% respectively with NRCC. In other words, the use of NRCC significantly declines the relocalization failures by more than 65% when the time interval exceeds two weeks.

Furthermore, Figure 15b shows the relocalization robustness at different areas. We conducted the experiment under the time interval of two weeks. The relocalizaiton failure rate at the office building, the gymnasium and the shopping mall is 2%, 5% and 10% with NRCC, compared to 4%, 15% and 25% when without NRCC. In average, NRCC declines relocalization failures by 57%. Especially in the gymnasium, the failure rate is decreased by 67%, which reflects remarkable improvement on robustness.

*4) Impact of Illumination:* We further tested our system with different illumination conditions in the office building, which undergos the most drastic illumination oscillation among the three areas. We first asked a leader to walk 4 pathways in the morning, afternoon and evening during a day. Then we asked volunteers to walk the same pathways correspondingly and calculate the navigation success rate of each test case. As shown in Figure 13, whenever followers walk the pathways, the navigation success rates are more than 80% and more than 90% if followers walk at evening. Generally, in the office building, we usually turn on all of the lights at evening, majority at morning but rarely at afternoon, which lead to the same video frame captured at evening enjoys the most drastic light and shadow oscillation. Therefore, the video frame has more ORB feature points than captured at morning and afternoon [26] and the relocalization success rate of the frame will increase.

*5) System Latency:* We recorded the time consumption of all frames in all followers' navigation experiments. As shown in Figure 14, *Pair-Navi* reduces the average relocalization time for one frame to 76ms. Compared with a complete SLAM system, the average delay is reduced by 17.4%, moreover, the percentage of frames using less than 100ms for relocalization increase from 81.6% to 91%. Surprisingly, we also observe that NRCC even slightly reduces the system latency by 4ms per frame in average. This is because NRCC shrinks the number of valid features points involved in relocalization. The average mask process time for each frame (resized as $640 \times 480$) is 0.18s in our system. In other words, a mask takes in charge of the NRCC of two frames that are captured in 0.2s. In a nutshell, *Pair-Navi* accomplishes relocalization and navigation within the system sampling time of 0.1s and runs fluently in real-time, with partial computation off-loaded to a cloud server. As our future work, we plan to optimize to a complete standalone system on smartphones based on model compressing.

*6) Energy Consumption:* We record the energy consumption of the follower's navigation app on the smartphones. On the Huawei P10, the program ran 41 minutes and 48 seconds and consumed 203.12mAh, while the battery level dropped from 100% to 69%. On the Lenovo Phab2pro, the program ran 44 minutes and 36 seconds and consumed 348mAh, while the battery level dropped from 100% to 85%. Since indoor pathways from one place to another are usually less than 15-min walking distance, we consider the energy consumption of *Pair-Navi* is acceptable.

## VI. Related Works

**Indoor P2P Navigation.** Traditional indoor navigation solutions require a global map of the indoor floor plan in infrastructure-ready indoor environments (e.g., Wi-Fi [31], [32], Bluetooth, RFID [5], etc.), and navigation instruction are provided based upon the absolute position in the global map. Recently, P2P navigation appears as another solution to indoor navigation, which does not rely on a complete global map and absolute localization in the map [8]–[10], [33]–[35]. In [35], an electronic escort system was proposed by using crowd encounter information and dead-reckoning techniques. The most relevant works *Travi-Navi* [10], *FollowMe* [9] and *ppNav* [8] all employ trace-driven navigation on smartphones. *Travi-Navi* synthesized Wi-Fi and inertial measurement to bootstrap navigation services without indoor floorplan. *FollowMe* exploited magnetic sensing and dead-reckoning to achieve lastmile navigation for smartphone users. *ppNav* utilized the ubiquitous Wi-Fi fingerprints in a novel diagrammed form and extract both radio and visual features of the diagram to track relative locations. In contrast, *Pair-Navi* exploits the power of vision, which is infrastructure-free and demonstrated to be more efficient, precise and further beneficial to various vision-based applications, such as indoor 3D-reconstruction, store sign identification, *etc.*

**Visual SLAM.** The pioneer work of monocular visual SLAM [12] adopted a filtering-based approach. Later, optimization-based methods [36] [13] came on stage and was demonstrated more accurate [37]. In recent years, *ORB-SLAM* [14], the state-of-the-art monocular visual SLAM work, used *DBoW2* [28] as the place recognition module, and *g2o* [38] as the optimization framework. Latest researches [20], [21] attempted to incorporate semantics into visual SLAM for better robustness in time-varying environments. Being able to compute the camera pose while generating the map and environment, visual SLAM is a suitable technique for indoor navigation.

**Instance Segmentation.** The Region-based CNN (R-CNN) approach [17] leveraged candidate object regions [39] and evaluated convolutional neural networks for each Region of Interest (RoI) for object detection. R-CNN was extended to allow RoI extraction on feature maps using RoIPool [40] and then advanced to Faster R-CNN [18] by learning the attention mechanism with a Region Proposal Network (RPN). On this basis, Mask R-CNN [19] was proposed to use mask predictions for classification and became the state-of-the-art in instance segmentation. More specifically, Mask R-CNN followed the idea of Fast R-CNN [40] that applies bounding-box classification and regression in parallel. In addition, it adopted a two-stage procedure like Faster R-CNN, with a first stage of RPN and a second stage of outputing a binary mask for each RoI.

## VII. Conclusion

In this paper, we present *Pair-Navi*, a robust and real-time P2P navigation system based on visual SLAM, requiring no pre-installed infrastructure or pre-deployed localization services. We implement *Pair-Navi* on commodity smartphones and conduct experiments in multiple buildings over two weeks. Experiment results show that our system outperforms existing solutions in navigation success rate and robustness. We believe *Pair-Navi* takes an promising step towards practical P2P navigation. Our future works target at fusing crowdsourced trajectories to generalize navigation routes to a larger scale and make a global consistent map.

## Acknowledgment

This work is supported in part by the National Key Research Plan under grant No. 2016YFC0700100, NSFC under grant 61832010, 61332004, 61572366, 61672319.

## References

[1] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye, "Push the limit of WiFi based localization for smartphones," in *Proceedings of ACM MobiCom*, 2012.

[2] C. Wu, Z. Yang, and Y. Liu, "Smartphones based crowdsourcing for indoor localization," *IEEE Transactions on Mobile Computing*, vol. 14, no. 2, pp. 444–457, 2015.

[3] C. Wu, Z. Yang, and C. Xiao, "Automatic radio map adaptation for indoor localization using smartphones," *IEEE Transactions on Mobile Computing*, vol. 17, no. 3, pp. 517–528, 2018.

[4] J. Wang and D. Katabi, "Dude, where's my card? RFID positioning that works with multipath and non-line of sight," in *ACM SIGCOMM*, 2013.

[5] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu, "Stpp: Spatial-temporal phase profiling-based method for relative rfid tag localization," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 596–609, 2017.

[6] M. Azizyan, I. Constandache, and R. Roy Choudhury, "Surroundsense: mobile phone localization via ambience fingerprinting," in *Proceedings of ACM MobiCom*, 2009.

[7] S. Liu and T. He, "Smartlight: Light-weight 3d indoor localization using a single led lamp," in *Proceedings of ACM Sensys*, 2017.

[8] Z. Yin, C. Wu, Z. Yang, and Y. Liu, "Peer-to-peer indoor navigation using smartphones," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1141–1153, 2017.

[9] Y. Shu, K. G. Shin, T. He, and J. Chen, "tmile navigation using smartphones," in *Proceedings of ACM MobiCom*, 2015.

[10] Y. Zheng, G. Shen, L. Li, C. Zhao, M. Li, and F. Zhao, "Travi-navi: Self-deployable indoor navigation system," in *Proceedings of ACM MobiCom*, 2014.

[11] Z. Yang, C. Wu, Z. Zhou, X. Zhang, X. Wang, and Y. Liu, "Mobility increases localizability: A survey on wireless indoor localization using inertial sensors," *ACM Computing Surveys*, vol. 47, no. 3, pp. 54:1–54:34, Apr. 2015.

[12] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proceedings of IEEE ICCV*, 2003.

[13] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Proceedings of IEEE ISMAR*, 2007.

[14] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[15] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[16] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *Proceedings of IEEE CVPR*, 2015.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of IEEE CVPR*, 2014.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of IEEE ICCV*, 2017.

[20] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *Proceedings of IEEE ICRA*, 2017.

[21] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. Montiel, "Towards semantic slam using a monocular camera," in *Proceedings of IEEE IROS*, 2011.

[22] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of IEEE CVPR*, 2013.

[23] "Ros kinetic kame," http://wiki.ros.org/kinetic.

[24] "Ros android," http://wiki.ros.org/android.

[25] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[26] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Proceedings of IEEE ICCV*, 2011.

[27] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[28] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proceedings of ECCV*. Springer, 2014.

[30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of IEEE CVPR*, 2016.

[31] K. Qian, C. Wu, Z. Yang, Z. Zhou, X. Wang, and Y. Liu, "Tuning by turning: Enabling phased array signal processing for wifi with inertial sensors," in *Proceedings of IEEE INFOCOM*, 2016.

[32] K. Qian, C. Wu, Y. Zhang, G. Zhang, Z. Yang, and Y. Liu, "Widar2.0: Passive human tracking with a single wi-fi link," in *Proceedings of ACM MobiSys*, 2018.

[33] T. H. Riehle, S. M. Anderson, P. A. Lichter, N. A. Giudice, S. I. Sheikh, R. J. Knuesel, D. T. Kollmann, and D. S. Hedin, "Indoor magnetic navigation for the blind," in *Proceedings of IEEE EMBC*, 2012.

[34] W. Storms, J. Shockley, and J. Raquet, "Magnetic field navigation in an indoor environment," in *Proceedings of IEEE UPINLBS*, 2010.

[35] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury, "Did you see bob?: human localization using mobile phones," in *Proceedings of ACM Mobicom*, 2010.

[36] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Real time localization and 3d reconstruction," in *Proceedings of IEEE CVPR*, 2006.

[37] H. Strasdat, J. M. Montiel, and A. J. Davison, "Visual slam: why filter?" *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[38] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "$g^2o$: A general framework for graph optimization," in *Proceedings of IEEE ICRA*, 2011.

[39] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 4, pp. 814–830, 2016.

[40] R. Girshick, "Fast r-cnn," in *Proceedings of IEEE ICCV*, 2015.