Smartphone-based Acoustic Indoor Space Mapping

SWADHIN PRADHAN, UT Austin, USA
GHUFRAN BAIG, UT Austin, USA
WENGUANG MAO, UT Austin, USA
LILI QIU, UT Austin, USA
GUOHAI CHEN, Network Technology Lab, Huawei, China
BO YANG, Network Technology Lab, Huawei, China

Constructing a map of indoor space has many important applications, such as indoor navigation, VR/AR, construction, safety, facility management, and network condition prediction. Existing indoor space mapping requires special hardware (*e.g.*, indoor LiDAR equipment) and well-trained operators. In this paper, we develop a smartphone-based indoor space mapping system that lets a regular user quickly map an indoor space by simply walking around while holding a phone in his/her hand. Our system accurately measures the distance to nearby reflectors, estimates the user's trajectory, and pairs different reflectors the user encounters during the walk to automatically construct the contour. Using extensive evaluation, we show our contour construction is accurate: the median errors are 1.5 cm for a single wall and 6 cm for multiple walls (due to longer trajectory and the higher number of walls). We show that our system provides a median error of 30 cm and a 90-percentile error of 1 m, which is significantly better than the state-of-the-art smartphone acoustic mapping system *BatMapper* [64], whose corresponding errors are 60 cm and 2.5 m respectively, even after multiple walks. We further show that the constructed indoor contour can be used to predict wireless received signal strength (RSS).

CCS Concepts: • **Human-centered computing → Mobile computing**; **Ambient intelligence**;

Additional Key Words and Phrases: Mapping, Smartphone, Acoustic Sensing, FMCW

ACM Reference Format:
Swadhin Pradhan, Ghufran Baig, Wenguang Mao, Lili Qiu, Guohai Chen, and Bo Yang. 2018. Smartphone-based Acoustic Indoor Space Mapping. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2, Article 75 (June 2018), 26 pages. https://doi.org/10.1145/3214278

1 INTRODUCTION

Motivation. There is a surge of interest in getting indoor structural information for a variety of purposes, such as indoor localization and navigation, network optimization, augmented reality (AR), virtual reality (VR). Specifically, an indoor map helps users to localize within indoors and to navigate inside the buildings. This is important since users spend approximately 80% of their time indoors [43]. However, such maps are rarely available, which becomes a major barrier in realizing ubiquitous indoor location based services (LBS). Additionally, thanks to recently released ARCore [3] and ARkit [4], there has been a rise in new AR applications, such as *Ikea Place* [10],

Authors' addresses: Swadhin Pradhan, UT Austin, USA, swadhin@cs.texas.edu; Ghufran Baig, UT Austin, USA, ghufran@cs.utexas.edu; Wenguang Mao, UT Austin, USA, wmao@cs.utexas.edu; Lili Qiu, UT Austin, USA, lili@cs.utexas.edu; Guohai Chen, Network Technology Lab, Huawei, China, chenguohai@huawei.com; Bo Yang, Network Technology Lab, Huawei, China, yanbo59@huawei.com.

2474-9567/2018/6-ART75 $15.00
https://doi.org/10.1145/3214278

Housecraft [9], AR MeasureKit [5] etc. These applications enable users to try out furniture, to achieve seamless home renovation, and to experience immersive games (which blends AR objects with the user's surrounding). These applications demand contextual structural information of the indoor space. In addition, home surveillance robots and autonomous home cleaning equipment require indoor maps to enhance their accuracy and coverage; and 360° videos also require indoor mapping to provide realistic and engaging experience. Furthermore, structural information allows us to accurately predict signal propagation and yield accurate estimation of wireless signals, which can be used for optimizing access points (AP) placement, selection, and rate adaptation. These applications call for a fast, low-cost, and easy-to-use indoor mapping system and can tolerate errors of a few centimeters.

The existing approaches generate indoor maps either manually or using specialized sensors (*e.g.*, laser rangers [57], depth cameras [41], sonars [29, 32]). The high cost of these approaches significantly limits the availability of indoor maps. Recently, some works [21, 26, 31, 34, 38] use crowd-sourcing to reduce deployment cost. However, crowd-sourcing incurs significant overhead and takes much longer to get the indoor maps. Meanwhile, it also raises incentive and privacy issues.

In terms of the types of techniques, cameras, LiDAR, and specialized sensors are often used for map construction. Vision-based approaches provide detailed indoor maps, but are computationally expensive, sensitive to lighting conditions and image quality, and raise privacy concerns. LiDAR are still costly and have trouble with transparent materials (*e.g.*, windows, glass doors), which is common in indoor settings. Microsoft Hololens [14], Google's Project Tango tablet [18], Oculus VR headset [15]) combine multi-camera and multi-depth sensors to improve accuracy. However, they are costly and have slow adoption [11, 13].

Our Approach. Inspired by many applications of indoor mapping and the lack of widely available tools, we develop a novel acoustic-based system for indoor map construction. It has several distinct benefits over the existing solutions: (i) it is low-cost and can be implemented on a smartphone without any extra hardware, and (ii) it is robust to ambient lighting conditions and transparent materials.

In our approach, we let a smartphone emit audio signals and analyze the signals reflected from the environment to infer the structure of indoor space. Our solution provides an infrastructure-free system to get the depth information by utilizing built-in speakers and microphones on smartphones. We develop a system, called SAMS (**S**martphone **A**coustic **M**apping **S**ystem), to get the indoor contour by just moving around while holding a smartphone. Specifically, SAMS applies Frequency Modulated Continuous Wave (FMCW) technology to estimate the distances to the nearby objects (*e.g.*, walls, doors, and shelves). We find that existing audio based tracking works [45, 48, 62] focus on getting only the shortest path or one moving path. The latter is achieved by finding the difference between consecutive samples to cancel out static multi-path. In comparison, we explicitly leverage multiple peaks in the FMCW profile to get critical structural information like corners or clutters, which are important for accurate map construction. Furthermore, we employ customized Inertial Measurement Unit (IMU) sensor based dead-reckoning combined with these distance measurements and systematic geometric constraints to derive contour of the surroundings in a calibration-free manner. We generalize our approach from a single wall setting to a multi-wall setting, account for clutters, and support both straight and curved surfaces.

As one can imagine, SAMS can readily help in enabling interesting applications like navigation for blind people (by detecting obstacles), enriching AR/VR applications, or helping in indoor construction (as illustrated in Fig. 1). SAMS can even provide semantic information of physical spaces (*e.g.*, corners, corridors) by analyzing reflected acoustic profiles, and furthermore, the mapping from SAMS can help in wireless signal strength prediction. We demonstrate this utility of our system by feeding the constructed indoor map to predict wireless received signal strength (RSS). We find that it yields accurate RSS prediction (within 1.5-2 dB error) and thus can help in application performance improvement. Another use-case might be to augment light-based distance estimation technique employed in Hololens or small LiDAR [1], in the cases of transparent or glass-like material in indoor. This acoustic based distance estimates will help these devices to create more accurate surrounding mapping templates.

(a) Detecting Obstacles for People. (b) Enriching AR/VR Application. (c) Helping in Construction Works.

Fig. 1. Possible Applications of Proposed SAMS System.

Our work is related to significant research on acoustic based motion tracking, gesture recognition, and activity detection [44, 47, 48, 50, 62]. In particular, *BatMapper* [64] system is the closest work to ours. It also uses acoustic signals to map the indoor space. However, *BatMapper* requires significant training to derive the parameters in its probabilistic algorithm, and requires multiple walks around the same space to generate the contour due to the underlying assumption. Furthermore, the paper does not specify how to construct the contours of the arbitrarily shaped surfaces. In comparison, our system eliminates *the need of training*, can support arbitrary contour and generate an indoor map by walking once around the space.

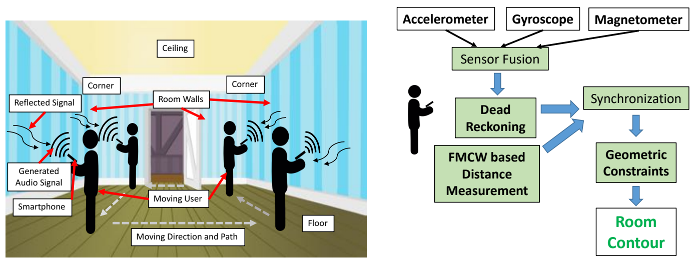
Our major contributions include: (i) applying FMCW-based distance measurement to contour construction by exploring its design parameters, studying its sensitivity under various environmental factors, and extracting features from FMCW profiles to identify the types of reflectors, (ii) realizing an infrastructure-free smartphone-based acoustic indoor mapping system that can support straight and curved surfaces, handle multiple objects, and automatically identify and remove clutters, and (iii) conducting extensive evaluation and applying the contour construction to predicting wireless signal strength. Our results show that SAMS significantly out-performs *BatMapper* and improves received signal strength (RSS) prediction.

Outline. The remaining paper is organized as follows. In Section 2, we overview our system. We present our detailed approach in Section 3. We describe the experiment setup in Section 4 and evaluation results with a comparison with *BatMapper* [64] in Section 5. We show the feasibility of predicting RSS based on the constructed indoor contour in Section 6. Finally, we review the related works in Section 7 and conclude in Section 8.

2 *SAMS*: OVERVIEW

In this section, we briefly outline how SAMS maps an indoor space. As illustrated in Fig. 2(a), a user walks around an indoor space while holding a smartphone in hand. SAMS running on the smartphone emits audio signals and analyzes the reflected signals to get the distance to nearby objects. Meanwhile, it estimates the user movement trajectory, and combines it with the distance estimation to create the contour of indoor space.

Fig. 2(b) illustrates the high-level ideas employed in SAMS. SAMS emits FMCW based audio chirps and analyzes the received signals to estimate relative distances between the phone and walls. It also identifies the corner and ceiling/floor, which are critical in forming the shape. Meanwhile, SAMS includes our dead-reckoning method [24, 51, 59]: it compensates for user's sway movement to get a correct user movement trajectory. Next, it synchronizes the measured distances to the nearby objects with the user trajectory using walking steps and turns them as synchronizing events. Afterwards, it removes outliers by applying geometric constraints and polynomial fitting techniques to get the indoor space contour. After a single walk around a room or indoor space, the user can accurately construct the indoor space. We further show that we can reduce the user profiling effort by using two microphones on the smartphone to simultaneously create contour of two walls in a corridor. SAMS is a *single-step* calibration-free fast indoor space mapping solution.



(a) SAMS in Action. (b) Block Diagram of SAMS.

Fig. 2. SAMS System.

3 SAMS

计算定位法

SAMS consists of three main modules: *distance estimation to nearby objects*, *dead-reckoning* module to determine the user trajectory, and *contour construction*. In this section, we describe each of these modules in detail.

3.1 Distance Estimation

闲碎的，不重要的

This module estimates the distance from the smartphone to the nearby objects (*e.g.*, walls, doors, shelves). We use FMCW to estimate the distance. Acoustic FMCW has been used in previous works (*e.g.*, [45, 47]), but those schemes were not designed for mobile and co-located microphone-speaker setting. Therefore, we perform non-trivial customization of FMCW based technique to improve accuracy in our settings. Below we first provide a brief overview of FMCW and then present our customization.

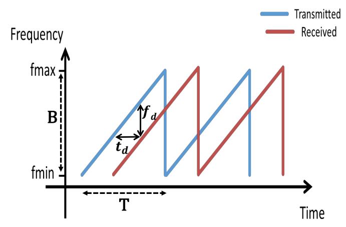


Fig. 3. FMCW Chirp Signal.

FMCW Primer. Frequency of an FMCW chirp increases linearly from f_{min} to f_{max} in each period, as shown in Fig. 3. The equation for the transmitted chirp is given as below where B is the bandwidth and T is the sweep time :

$$S_t(t) = cos(2\pi f_{min}t + \frac{\pi B t^2}{T})$$

信号衰减导致了接收信号的振幅衰弱

When the transmitted chirp signal is reflected by the object and arrives at the receiver after a delay t_d, the received signal, $S_r(t)$ is attenuated and delayed version of the transmitted signal, so it becomes $S_r(t) = \alpha S_t(t - t_d)$ where α is the attenuation factor. The receiver mixes (*i.e.*, multiplies) the received signal ($S_r(t)$) with the transmitted signal ($S_t(t)$), and gets $S_m(t) = S_r(t) \times S_t(t)$. If the distance between the object and smartphone is $\frac{R}{2}$, t_d is given by R/v_s, where R is the two-way range estimate and v_s is the speed of sound. Plugging t_d into the above equation and after simplifying, $S_m(t)$ becomes:

$$S_m(t) = \alpha cos(2\pi f_{min}\frac{R}{v_s} + 2\pi Bt\frac{R}{v_sT} - \pi B\frac{R^2}{v_s^2 T}))$$

.

Analyzing the frequency spectrum of the mixed signal, we have $f_p = \frac{1}{2\pi}\frac{\delta Phase}{\delta t} = \frac{BR}{v_s T}$. This means that the frequency of the mixed signal peaks at $\frac{BR}{v_s T}$. Based on the observed f_p in the FFT spectrum, the distance R can be derived as: $R = \frac{f_p v_s T}{B}$. If there are multiple propagation paths between the transmitter and receiver, multiple peaks are observed in the FFT spectrum of the mixed signal. After converting the x-axis to distance, we generate FFT spectrum and call it a *FMCW profile*.

3.1.1 Our FMCW. We develop a customized FMCW module as shown in Fig. 4. The speaker transmits an FMCW chirp periodically and the microphone records the reflected chirps simultaneously. This module employs correlation based synchronization technique described in [45] to identify the start of the signal. Then, applying FFT on the multiplicative mixture of generated and received signal, we get the distance estimates. In the following, we explain: (i) how to choose FMCW parameters, and (ii) how to select peaks for accurate and fast distance estimation.

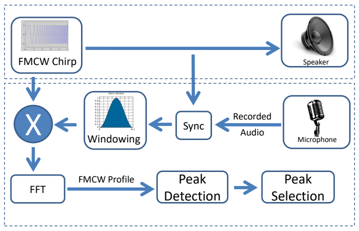
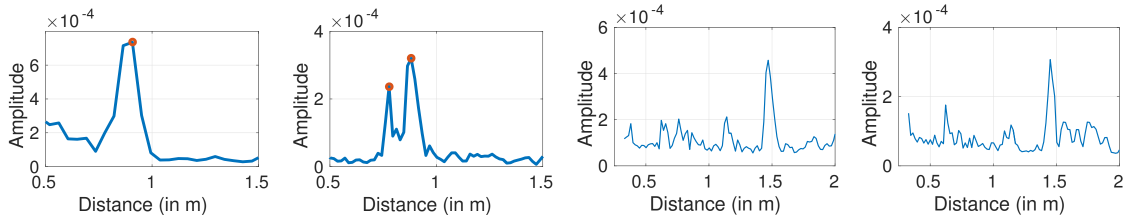


Fig. 4. Our FMCW Processing Module.

Choosing FMCW Parameters. The performance of FMCW depends on two important parameters: bandwidth and chirp duration. Bandwidth determines the FMCW resolution, and chirp duration determines the range.

Resolution is defined as the minimum distance between two objects that can be differentiated. It is important because the peaks corresponding to the objects of interest may be merged if the objects are close enough. The merged peak becomes wider and causes erroneous distance estimation. Fig. 5(a) and Fig. 5(b) show how resolution is dependent upon the transmission bandwidth. Fig. 5(b) shows that a 10 KHz signal can easily differentiate reflectors that are separated by 10 cm, and Fig 5(a) shows that a 4 KHz signal cause multiple peaks to be merged into a single wider peak. We find that a resolution of 10 cm is good enough for clutter removal in our experiments. So, we use a FMCW chirp of bandwidth 10 KHz: from 11 to 21 KHz. For comparison, we have also evaluated with a smaller bandwidth: from 18 to 22 KHz (also in sub-section 5.9).



(a) FMCW Profile for 4KHz Chirp. (b) FMCW Profile for 10 KHz Chirp. (c) Reflectors on Sides (60 cm) and in Front (150 cm). (d) Reflectors Behind (60 cm) and in Front (150 cm).

Fig. 5. FMCW Profiles.

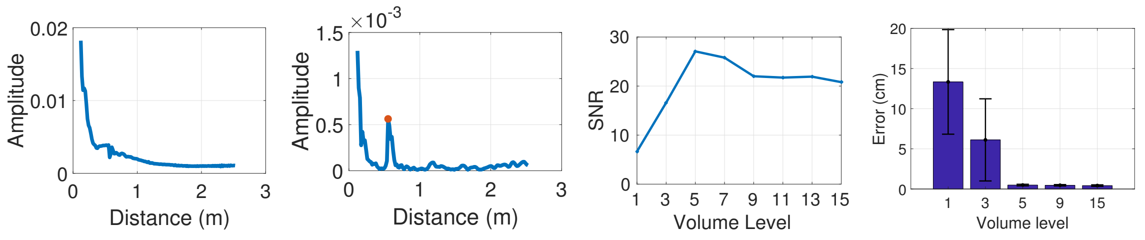
A longer chirp allows us to measure a longer distance using FMCW. However, while a user is walking, a longer chirp duration will introduce more error since the position at which the signal is transmitted and received is different. Therefore, we should choose an appropriate chirp duration to support large range and responsiveness to user movement. Through measurement studies, we find that 30 ms chirp duration works well. This time-period selection gives us a range of 346 m $\times 0.03 = 10.38$ m two-way and 5.19 m one-way. This sampling rate is sufficient for a walking speed (*i.e.*, within 2 m/s).

FMCW Peak Selection. Multi-path propagation and noise may cause multiple FMCW peaks. It is important to select the right peak for distance estimation. We approach this issue by (i) improving the peak-to-side-lobe ratio through windowing, and (ii) identifying the characteristics of correct peaks to be used for distinguishing from the wrong peaks. Below we elaborate.

(i) *Improving peak-to-side-lobe ratio.* We multiply the received signal with Hanning window [29] to improve the peak-to-side-lobe ratio:

$$H[n] = 0.5 * \left(1 - \cos\left(\frac{2*\pi*n}{N-1}\right)\right)$$

Since the amplitude of the direct path is orders of magnitude higher than the reflected signals, the side lobes make it almost impossible to detect peaks from reflected signals in the FMCW profile. As shown in Fig. 6, where only the side lobes from the direct path are present without windowing. To understand the impact of using Hanning window, we have performed 56 measurements in 4 different indoor settings (lab, office, meeting room, corridor) as the distances varies from 0.5 m to 4 m. We have observed that the side lobes completely overwhelm the peaks from the objects when the distance is within 2.0 m, which results in an error of over than 95 cm. For distances greater than 2.0 m, the side lobe peaks are still comparable to reflected peaks, so it is still quite likely that a wrong peak is selected. Therefore, windowing the signal to reduce the side-lobes is crucial for its operation.



(a) Without Windowing. (b) With Windowing. (c) SNR. (d) Distance Estimation.

Fig. 6. (a) No Clear Peak in the FMCW Profiles Without Window. (b) Windowing Yields Clear Peak. (c) SNR under Different Speaker Volume Level. (d) Distance Estimation Error under Different Volume Level.

(ii) *Identify correct peak.* We find the maximum peak observed in the FMCW spectrum corresponds to the object that is directly in front of the speaker irrespective of the distances to other reflectors. This can be seen in Fig. 5(c) and Fig. 5(d), where the maximum peak is always at 1.45m, corresponding to the distance to the reflector in front. The magnitude of this peak is considerably high even when the reflectors at the side or at the back are only 0.6 m from the phone. This is because the main lobe of the phone speaker, which contains the maximum energy, is always at the front as observed in [49]. Using this observation, when there is a clear peak, we select the distance corresponding to the *maximum* peak and use it for our contour creation algorithm. However, in certain scenarios (*e.g.*, due to user's non-deterministic movement, change in phone orientation etc.), there may be multiple peaks in our FMCW profile with the maximum peak not being the one from the object in front of the phone speaker.

To address this issue, we use a peak selection algorithm described in Algorithm 1 to select the peak that most likely comes from the object in front of the phone. We do this for the set of FMCW profiles between each turn.

We group the peaks in each FMCW profile based on their proximity to the previous peaks. A peak is assigned to a group if the distance between this peak and the last peak assigned to the group is within a threshold T (line 7 - 11). If no such group is found a new candidate group is added with this peak (line 12 - 14). At the end, the group with the maximum number of peaks assigned to it is chosen for estimating distance used in contour construction (line 15). Algorithm 1 summarizes our peak selection algorithm. The threshold T has been selected as 10 cm.

Algorithm 1 Peak Selection

```
1:  procedure PEAKSELECTION
2:  # S is set of range values from FMCW
3:      $Candidates \leftarrow$ EMPTYVECTOR
4:      $NumCand \leftarrow 0$
5:      for $r_t \in$ S do
6:          $Assigned \leftarrow False$
7:          for $i = 1; i \leq NumCand; i++$ do
8:              $lastVal \leftarrow Candidates[i][end]$
9:              if $|r_t - lastVal| < thresh$ then
10:                 $Candidates[i] \leftarrow$ APPEND( Candidates[i], $\mathbf{r_t}$ )
11:                 $Assigned \leftarrow True$
12:         if !$Assigned$ then
13:             $NumCand \leftarrow NumCand + 1$
14:             $Candidates[NumCand] \leftarrow$ APPEND( Candidates[NumCand], $\mathbf{r_t}$ )
15:     ret $\leftarrow maxLengthVector(Candidates)$
16:     return ret
```

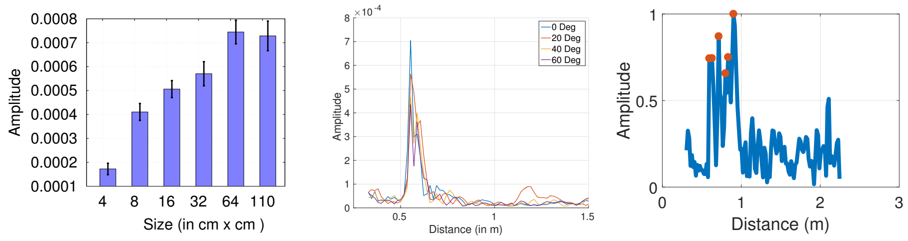
3.1.2 Observations on Our FMCW. We make a few important observations from our FMCW experiments. We will use these observations for contour construction.

Volume. The magnitude of FMCW peaks depends on the speaker volume. Very low volume makes FMCW peaks hard to detect, while too high volume produces loud audible noise and causes signal distortion, which may actually reduce SNR (Signal-to-Noise Ratio)[1] value. Therefore, we need to select an appropriate volume level to balance these factors. To find an appropriate volume level, we collect 100 distance measurements at different volume levels using our samsung S7 smartphone (which has 15 levels). The distance is measured from a wall at distances between 0.5 to 4 m in different indoor locations. Average SNR values for different volume levels are shown in Fig. 6(c). In our experiments, we have found that the volume level 5 gives us the maximum SNR. We also show in Fig. 6(d) that the distance estimation performance for the volume levels 5 and above, are similar in our range of interest (up to 4 m). However, for a volume level below this, the reflected peak gets very weak for the distances over 2 m, resulting in high median error (around 5 cm). Therefore, we use volume level 5, *i.e.*, 30% of the highest volume level for our evaluation. This volume level gives a barely audible signal for frequencies above 11 KHz with similar performance as the highest volume level.

FMCW Peak vs. Object Size. Fig 7(a) shows the amplitude of the reflected signal increases with the size of the reflector, and tapers off after a certain size. This means that the microphone only receives reflection from a certain area of the reflector since reflections from the other part of the reflector would not reach the microphone at the bottom of the phone. This explains why we see sharp peaks even when we have a relatively large reflector

[1]We calculate SNR by taking the ratio of peak amplitude of the reflected peak in FMCW spectrum with the average of the whole FMCW spectrum.

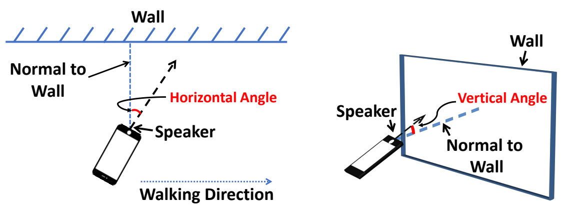
(*e.g.*, wall). Otherwise, we could have received reflections from many more points on the reflector and get very wide FMCW peaks, which would be difficult for accurate distance estimation.



(a) FMCW Peak Amplitudes vs. Different Object Sizes. (b) A FMCW Profile for Smartphone Placed in Front of a Wall. (c) An Example FMCW Profile Observed at a Corner.

Fig. 7. FMCW Peak and Profile Characteristics.

Perpendicular Distance. We place the smartphone in front of a wall at a distance of 60 cm, and measure distance using FMCW while placing the smartphone at different horizontal angles with the normal of the wall. Zero degree is when the speaker and microphone are facing directly towards the wall. As shown in Fig. 7(b), regardless of the angle of the smartphone with the wall, we always get the distance that is perpendicular to the wall. This is because only the signal reflected from that region comes back to the microphone. This is a *critical observation* that we use in constructing a contour.



(a) Horizontal Angle. (b) Vertical Angle.

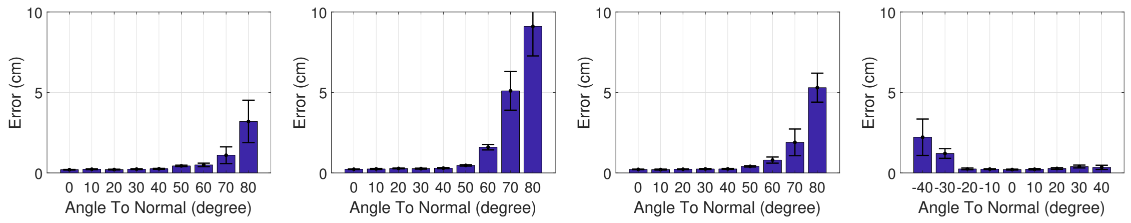
Fig. 8. Different Types of Angle in Our Experiments.

To verify this observation, we measure distances to a large reflector when placed at different angles. We have considered both *horizontal* and *vertical* angles with the normal of the wall for our experiments, as depicted in Fig. 8. We have performed the experiments in 4 different indoor settings (*e.g.*, large meeting room, open office space, corridor and lab). In each setting, we take 10 measurements for each angle as the distance is varied between 0.4 to 4 m.

In Fig. 9(a), we evaluate the distance estimation error in a corridor and lab, where a wall is the only reflector. The distance estimation error remains similar until the phone is kept at more than 80 degree horizontal angle. This happens because the reflected peak is too weak. Fig. 9(b) further shows the results at an office and meeting room where there are other objects (*e.g.*, chair, board) and the major reflector is still a wall. In this case, we start to see reflected peaks from other reflectors when the angle is greater than 30 degrees. However, we are still being able to identify them as separate peaks; furthermore, the peak selection module ensures that the correct peak is chosen, resulting in a negligible error. However, for angles greater than 60 degrees, the reflected peak from the wall is often missed, because it is much weaker than the reflection from other objects in front of the speaker, which yields larger error. Next, we evaluate using concave or convex walls made of wooden sheets. As shown in Fig. 9(c), the results are similar to that of straight walls.

Fig. 9(d) shows the distance estimation when the orientation of a smartphone is changed vertically. A negative angle indicates that the phone is tilted downwards. The result shows that the performance for measuring perpendicular distance of vertical angle are similar to those of horizontal angles, except when the negative angles is below −30 degree, the error is significantly high. This is because the phone points towards the intersection between floor and wall causes merged peaks, which is also shown in Fig. 7(c).

These results confirm the observation that irrespective of the holding angle, SAMS measures the perpendicular distance to the wall for a wide range of vertical and horizontal angles. Furthermore, it demonstrates the robustness of our system under various phone orientation.



(a) Perpendicular distance error in the environments with single reflector. (b) Perpendicular Distance Error in the Environments with Multiple Reflectors. (c) Perpendicular Distance Error for the Curved Surfaces. (d) Perpendicular Distance Error for Different Vertical Angles.

Fig. 9. Error in Estimating Perpendicular for Different Orientations.

FMCW Profile of an Intersection. Reflections from an intersection (*e.g.*, the perpendicular wall intersections) give us a very distinct profile compared to reflection from a straight wall. As shown in Fig. 7(c), the FFT of signals reflected from a corner has a wide peak with multiple small peaks in the neighborhood. This is because audio signals going through $1st$, $2nd$, or even higher order reflection can still reach the microphone. Since the distance of these paths are similar, the resulting peaks merge and the merged peak makes it difficult to obtain accurate distance estimation. Instead of trying to extract accurate distances from these samples, we use this observation to detect corners and remove these samples from contour detection. Using these observations, we denote an FMCW reading correspond to a corner if there exist multiple peaks with magnitudes comparable to the maximum in the neighborhood (within 50 cm) of the maximum peak. Algorithm 2 shows the pseudo code for corner detection using the FMCW profile. To find an appropriate value for the threshold of the number of such peaks in Algorithm 2, we have collected around 200 traces from the corners in multiple indoor locations in different buildings. We run Algorithm 2 on these traces and traces of reflections from walls in different indoor settings for different threshold values. The results are summarized in Table 1, where false positives are straight walls that are classified as corners and false negatives are corners that are not classified as corners. Using these results, we set $Thresh_1 = 4$ for corner detection since it minimizes both false positives and negatives.

We also observe that the intersection between a wall and floor or between a wall and ceiling also introduces additional peaks. To confirm this observation, we have done an experiment in an an-echoic chamber as described later. We observe an additional peak whenever the floor reflector is introduced. Fig. 10 shows an additional peak in FMCW profile, which is absent from Fig. 11. Using the distances to the wall and from floor to hand, the position of an intersection can be easily estimated using Pythagoras theorem. At the start of our experiment, we measure the floor to hand distance by pointing the phone speaker towards the floor. Now if we see a peak with a distance close to the intersection between the floor and wall (which is $\sqrt{wall\_dist^2 + floor\_dist^2}$), we consider the peak is from the intersection and ignore it. Otherwise, it is considered to be from another wall and peak selection algorithm decides which of the remaining peaks to chose for contour construction.

Algorithm 2 Corner Detection, $Thresh_1$ (number of peaks in the neighborhood) and $Thresh_2$ (spread of the magnitude of the peaks in neighborhood) are set according to the measurement results to 4 and 0.6, respectively.

```
1: procedure CORNERDETECTION
2:     w ← PEAKS WITHIN 0.5M OF MAX. PEAK IN FMCW PROFILE
3:     if SIZE(w)) ≥ $Thresh_1$ then
4:         if STDEV(w) < $Thresh_2$ then
5:             CORNER ZONE DETECTED
```

Table 1. Impact of the Threshold Values on Corner Detection.

Thr. (# pks.)	FP (%)	FN (%)
2	82	0
3	34	3
4	5	7
5	2	46

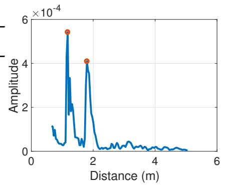


Fig. 10. FMCW Profile With the Floor.

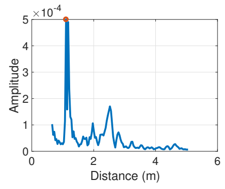
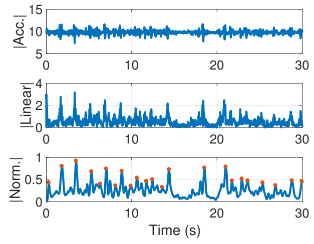


Fig. 11. FMCW Profile Without the Floor.

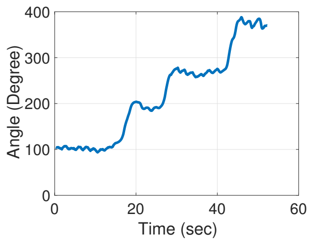
3.2 Dead Reckoning

Next we need to estimate the user's trajectory using smartphone IMU (Accelerometer, Gyroscope, and Magnetometer). Our goal is to extract the following information correctly: step size estimation, step count, and heading direction. We assume that the phone's heading direction as the user's heading direction, so we do not need to compensate for the phone's orientation. This assumption can be removed if we integrate recent development in heading direction estimation [51].

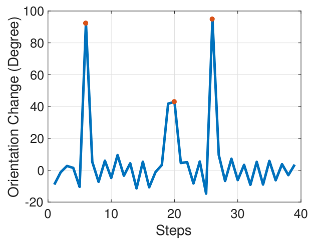
Our dead-reckoning module selects an ensemble of different techniques proposed in literature by keeping the system robust and no need for training. Note that despite significant existing works, accurately estimating the orientation during walk remains open [2, 56, 65]. We develop a simple yet effective cascade-based compass (accelerometer and magnetometer) and gyroscope aided technique to determine the orientation during walk. We employ frequency-based noise filtering to minimize noise introduced during walk. To enhance the accuracy of step size estimation, we carefully weigh the vertical component of acceleration change after employing frequency-based low-pass filter.



(a) i. Raw Acceleration Output. ii. Smoothed Linear Acceleration. iii. Normalized Linear Acceleration.



(b) Heading Direction Estimates in a Three-turn Square Walk.



(c) Gyroscope Data Trend in a Three-turn Square Walk.

Fig. 12. Dead-reckoning in Action.

Step Detection. Since the maximum jump in accelerometer reading occurs when the heel strikes the ground [51, 54, 65], we devise an algorithm to identify peaks for step detection. Then, we get the magnitude of the 3-axis linear acceleration (after removing gravity component) (*i.e.*,

$$A = \sqrt{LinearAcc_X^2 + LinearAcc_Y^2 + LinearAcc_Z^2}$$

), as shown in the second part of Fig. 12(a). We feed the signal magnitudes through a smoothing filter in the pre-processing step. To extract a better low-band step component, a low-pass filter is then applied to filter out high-frequency accelerations caused by the phone's random movement [54]. Next, we normalize the filtered acceleration values. After the low-pass filter, a peak recognition algorithm with a sliding window is used to detect peaks in the filtered data. Specifically, a_i is detected as a peak (*i.e.*, a user step) if it is larger than all samples located in the range of $[t(i) - t_w/2, t(i) + t_w/2]$. Since the user step frequency is in general lower than 3Hz, the window size t_w in the current implementation is set to 0.3s. Fig. 12(a) shows the acceleration signals after post-processing. In this figure, the first row is the original acceleration output from the smartphone, and the second and third rows are smoothed linear acceleration data without gravity component and the corresponding low-band component, respectively. Detected peaks are highlighted in the third row in red dots.

Step Size Estimation. There have been a series of works [20, 23, 46, 53] that estimate the step size based on extensive training. To avoid personalized training, we assume that a user moves in a similar speed so that step size remains relatively constant. We use a calibration-free method for step size estimation based on the following formula $0.98 \times \sqrt{A_{max} - A_{min}}$ (as derived in [53]), where A_{max} and A_{min} are the maximum and minimum linear acceleration in a step duration. For every step detected, we calculate the step-size. We have observed the median error is 6cm across 3 users over 60 trials.

Heading Direction Estimation. To perform dead-reckoning, we need to know the heading direction of the user. We combine gyroscope and magnetometer to get a heading direction estimate. For our case, we first convert both sensor values to the global coordinate system by multiplying them with the rotation matrix, and then use $0.97\times$ *Orientation from Gyroscope*$+0.03\times$*Orientation from Magnetometer* to extract the heading direction. A sample code is given in [19]. However, we have observed that the compass shows erratic variation due to ferromagnetic interference, which results in a variance of more than 120 degree within 3 seconds. In such scenarios, we only use orientation from gyroscope reading. Fig. 12(b) shows the heading direction estimate for a trajectory where the user moves in a rectangle making three turns and moving in a straight line in between. It shows that even though we can clearly identify turns as large jumps in orientation there is an error of up to 20 degrees when moving in a straight line. Median error for 100 instances of heading direction estimate is 10 degree and 90th percentile error is 25 degree. This happens due to small variation in walking pattern changes or subtle hand sways and environmental magnetic interference. To eliminate this inherent error, we consider this small variation in heading direction estimates as noise. Therefore, SAMS only tries to identify large changes in the heading direction, which we identify as the *turns* (shown as red dots in Fig. 12(c)). We assume a user makes only 90 degree turns and straight movement between the two turns to avoid heading direction error. The same assumptions are used in *BatMapper* system [64]. This is not a significant limitation since most buildings, rooms, corridors are rectangular. To evaluate the accuracy of our heading direction estimation, we have collected IMU traces for over 100 walks of 3 users with multiple turns having straight segments in between. The traces were collected in many indoor environments (*e.g.*a large room, a lab, an open office space, a corridor and a classroom). These areas tend to have high ferromagnetic interference due to the presence of HVAC system installed. We are able to detect all turns with a 100% accuracy.

Summing up, we detect steps by selecting peaks in processed accelerometer data and identify *turns* to get heading direction. Then, we use this orientation estimation and step counts to construct the path necessary for our contour construction.

3.3 Contour Construction

Algorithm 3 Wall Association, where $Thresh = 0.1, wnd = 5, N = 3$ work well in all the scenarios we have tested.

```
procedure WALLASSOCIATION
# S is set of phone coordinates from dead reckoning and range values from FMCW
    $wallID \leftarrow 0$
    $counter \leftarrow 0$
    $\mathbf{w} \leftarrow$ EMPTYVECTOR
    for $(x_t, y_t, r_t) \in$ S do
        $g_{prev} \leftarrow$ EMPTYVECTOR
        $g_{aftr} \leftarrow$ EMPTYVECTOR
        for $i = 1; i \leq wnd; i++$ do
            $g_{prev} \leftarrow Gradient((x_t, y_t, r_t), (x_{t-i}, y_{t-i}, r_{t-i}))$
            $g_{aftr} \leftarrow Gradient((x_t, y_t, r_t), (x_{t+i}, y_{t+i}, r_{t+i}))$
        if STDEV$(g_{prev})$ − STDEV$(g_{aftr}) > Thresh)$ then
            $counter \leftarrow counter + 1$
            if $counter > N$ then
                $wallID \leftarrow wallID + 1$
                for $j = 0; j < Thresh; j++$ do
                    $\mathbf{w} \leftarrow ((x_{t-j}, y_{t-j}, r_{t-j}), WallID)$
                $counter \leftarrow 0$
        else
            $counter \leftarrow 0$
    return w
```

After getting the user trajectory and the corresponding distances to the nearby objects during the walk, next we construct a contour of an indoor space. This involves (i) associate distance measurements to each object (*e.g.*, wall), (ii) estimate the distance to the object, (iii) construct a contour based on the distance estimation. Below we describe these steps in details.

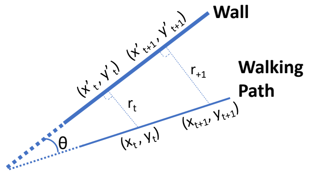


Fig. 13. Geometric configuration.

Wall Association. We associate each measurement (x_t, y_t, r_t) from dead-reckoning and FMCW modules to a wall in the room. The algorithm for wall assignment is summarized in Algorithm 3. For the straight walls in the room, the gradient associated with all points on a wall should be similar. At the intersection, as we transit between walls, the gradient changes. We assign all points with similar gradients to the same wall. From Fig. 13, if we have two sets of readings (x_t, y_t, r_t) and $(x_{t+1}, y_{t+1}, r_{t+1})$, we can get gradient of the path as $m_1 = \frac{y_{t+1} - y_t}{x_{t+1} - x_t}$. Then the gradient of wall m_2 can be calculated as $m_2 = tan(\theta + tan^{-1}(m_1))$, or $m_2 = tan(-\theta + tan^{-1}(m_1))$ depending on which side of the path wall is present. This can be determined using the phone orientation from the software based compass as described in section 3.2, as the user would be pointing the phone towards the wall while moving. Using similar triangle property and trigonometry, θ is calculated as

$$\theta = sin^{-1}\left(\frac{r_{t+1} - r_t}{\sqrt{(y_{t+1} - y_t)^2 + (x_{t+1} - x_t)^2}}\right)$$

For each point, we get a set of gradients g_{prev} from wnd number of points that were observed before and another set of gradients g_{aftr} from wnd number of points that were observed after that point (line 9 - 11). If all the points are on the same wall, they have similar gradients. However if some of the points are on a different wall, the estimated gradients will be different. This means that if the difference between g_{prev} and g_{aftr} is significant, it is the start of a new wall. If we find N such points together, we assign all of them to a new wall (line 12 - 18), otherwise they are marked as potential outliers to be removed later during clutter filtering (line 20).

However, the above assumptions may fail when the wall has a curved surface. In this case, the gradient changes slowly instead of abrupt gradient change at corner between two straight lines. We experimentally determine this threshold using traces from different curved wall.

Wall Co-ordinate Estimation. After wall association, next step in room reconstruction is to estimate the actual coordinates (x'_t, y'_t) on the wall. Since we always measure the perpendicular distances from the wall as mentioned in 3.1.2, we have the following two geometric constraints on the values of (x'_t, y'_t).

$$(y'_t - y_t)^2 + (x'_t - x_t)^2 = r_t^2$$

.

$$\frac{y'_t - y_t}{x'_t - x_t} = \frac{-1}{m_2}$$

where m_2 is the gradient of wall (or tangent to the wall if it is a curved surface) and is calculated as the average of the gradients of the L consecutive points assigned to the same wall excluding the outliers. A large value of L would average out the error in the m_2 for straight walls but would increase the error for curved surfaces as the line through those points would not correspond to the tangent. We pick $L = 3$ as it gives us reasonable accuracy for both straight and curved surfaces. Using these constraints, x'_t and y'_t can be calculated as

$$x'_t = \pm\sqrt{r_t^2 \frac{m_2^2}{m_2^2 + 1}} + x_t;$$

$$y'_t = (x_t - x'_t)/m_2 + y_t$$

This would potentially give us two set of values for each x'_t and y'_t. Among these two sets, we pick the set of values for which the gradient estimates are closer to m_2 In practice, we pick the set that gives us the average gradient closest to m_2.

Outlier Removal. After associating points with each of the walls and generating point coordinates on the wall, we identify the outliers by performing linear regression on the constructed points (having the same wall id) and applying Cook's method [8, 28] on those points. Cook's distance of each point is commonly used to estimate the influence of a data point when performing a least-square based linear regression. We identify the point as an outlier and remove them from the group if its Cook's distance (D_i) is greater than certain value. More specifically, if $D_i > 4/n$ [28], where n is the number points in that linear regression window, we perform that outlier removal operation. We have observed that these outliers are usually due to hand sway movement and clutter. We select the size of n as 5 such that linear regression can be performed on small segments (within a reasonable walking speed) during outlier detection. A smaller n degrades the accuracy of curved surface construction since we remove points which should have been on the surface. A larger n degrades the accuracy of clutter removal since we will consider a clutter as part of the surface. Our choice of n strikes a good balance in finding accurate surface contour in presence of clutter and removing wrong distance estimates. Note that, as any curved surface can be approximated using multiple small line segments, outlier detection operation is performed without any degradation of curved surface estimation.

Contour Construction. The final step is to reconstruct actual contour from generated coordinates. Linear regression based contour construction is preferred if the wall is straight; otherwise spline interpolation should be used. To determine whether it is a curved or a straight wall, we compare the root-mean-squared-error (RMSE) between the linear regression ($RMSE_{lin}$) and spline interpolation ($RMSE_{spline}$), where $RMSE$ is defined as the sum of distance differences from each measured point associated with the wall to its closest point on the fitted line. We consider the wall as curved if $RMSE_{lin} - RMSE_{spline} \geq T$, and otherwise consider it as a straight wall.

The value of threshold T is critical in this classification. To find an appropriate threshold, we construct different curvatures using a wooden sheets between two straight walls and run our classifying technique on measured traces. We observed that for $T \leq 8$ cm we can always detect a curved wall with a 100% accuracy. Since using a linear fit for a curved wall results in larger error from ground truth than choosing a spline interpolation for straight wall, we use $T = 8$ cm for our experiments. In all our experiments, we observe that spline interpolation is used for all curved surfaces and only less than 2% straight walls. The points of intersection between different consecutive walls give the estimated coordinates for the corners. Once we have equations of the walls and coordinates of the corners, we can plot the room contour.

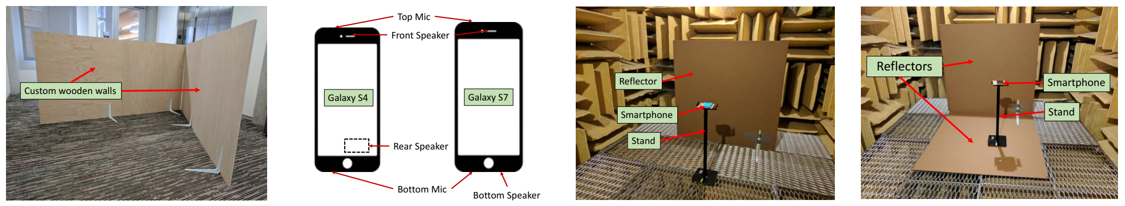
In summary, to build a contour using SAMS, a user moves along the walls at a normal walking speed keeping the speaker of the smartphone facing the wall. As illustrated in Fig. 9, the system is shown to be resilient to the speaker orientation.

3.4 Final Algorithm

Based on the above observations, we present the pseudo-code of SAMS algorithm in 4. After getting the distance estimation of dead-reckoning module at every T_w interval (which is 0.2 s for our experiments), we combine them with the FMCW based distance measurement while using steps as synchronization points. Next, we apply the *Wall Association*, *Co-ordinate estimation*, and *Outlier Removal* algorithm to create contour.

4 SETUP

In this section, we describe our experimental setup in detail. For our experiments, we use Samsung Galaxy S7, which has a speaker at the bottom and two microphones (mic) with one at the top and one at the bottom (Fig. 14(b)). In most of the experiments, we use the audio recorded by the bottom mic since the top mic is mostly used for noise cancellation [6]. Furthermore, the quality of the audio recorded is dependent upon the speaker position (*e.g.*, if the speaker is at the bottom, the recorded audio quality at the top mic is low). We also perform experiments with Samsung Galaxy S4 (Fig. 14(b)) to reduce user effort by simultaneously using both mics.



(a) Contour Experimental Setup With Wooden Walls. (b) Layouts of Smartphones Used. (c) Experiment for Understanding FMCW Based Distance Measurement. (d) Experiment for Understanding the Impact of Junction.

Fig. 14. Experimental Setup, Phone Layouts, and An-echoic Chamber Setups.

Algorithm 4 SAMS

```
procedure getFMCWDistance
    $t_0 \leftarrow$ currentSystemTime
    $t \leftarrow t_0$
    $\mathbf{w} \leftarrow$ emptyVector
    while $t \leq t_0 + T_w$ do
        $t \leftarrow$ currentSystemTime
        $\mathbf{w} \leftarrow$ append($\mathbf{w}$, FMCWDistance)
    return $\mathbf{w}$
procedure getDeadReckoningDistance
    $t_0 \leftarrow$ currentSystemTime
    $t \leftarrow t_0$
    $\mathbf{u} \leftarrow$ emptyVector
    while $t \leq t_0 + T_w$ do
        $t \leftarrow$ currentSystemTime
        $\mathbf{u} \leftarrow$ append($\mathbf{u}$, stepDistance)
    return $\mathbf{u}$
procedure getOrientation
    $t_0 \leftarrow$ currentSystemTime
    $t \leftarrow t_0$
    $\mathbf{v} \leftarrow$ emptyVector
    while $t \leq t_0 + T_w$ do
        $t \leftarrow$ currentSystemTime
        $\mathbf{v} \leftarrow$ append($\mathbf{v}$, CompassGyroOrientation)
    return $\mathbf{v}$
procedure createContour
    $\mathbf{s} \leftarrow$ emptyVector
    $\mathbf{fd} \leftarrow$ getFloorDistance
    while True do
        $\mathbf{w} \leftarrow$getFMCWDistance
        $\mathbf{x} \leftarrow$getDeadReckoningDistance
        $\theta \leftarrow$getOrientation
        if detectNotCorner then
            removeJunction($\mathbf{fd}$);
            wallAssociation;
            $\mathbf{l} \leftarrow$co-ordinateEstimation
            removeOutliers($\mathbf{l}$);
            $\mathbf{s} \leftarrow$ append($\mathbf{s}$, $\mathbf{l}$)
```

To perform the experiments, the user generally moves around the indoor space holding the phone in a straight line while the phone microphones face the walls. We develop an android application that runs on the smartphone to play audio chirp, record reflected audio signal and different IMU sensor data simultaneously. The recorded audio signal and sensor readings are processed in MATLAB to get the final contour. We first try to understand the performance of SAMS for arbitrary contours using controlled experiments, with increasingly challenging configurations. We use 4 ft by 6 ft wooden structures shown in Fig. 14(a) to create different corners, straight or curved surfaces, and closed shapes like a rectangle etc. Once we have an understanding of how the system works for arbitrary contours, we then test the system by mapping the real rooms, corridors, and a complete floor layout of a real building. We use laser based system [12] to get the ground truth. The system transmits distance measurements to the smartphone via bluetooth and is accurate to 1.5 mm.

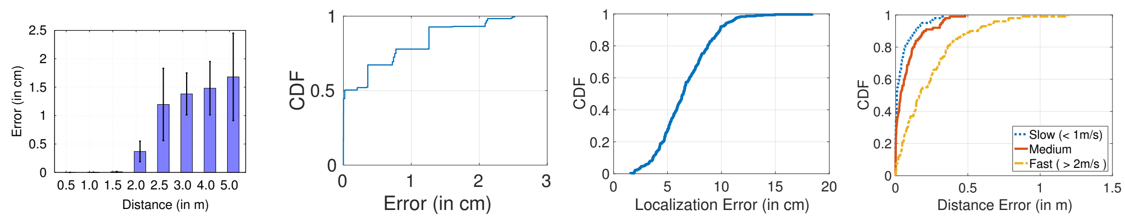
Furthermore, to better understand the audio reflection patterns, we also have done experiments in an-echoic chamber as shown in Fig. 14(c) and Fig. 14(d) . Fig. 14(d) shows the smartphone and different reflectors in the an-echoic chamber.

5 EVALUATION

In this section, we evaluate our system SAMS in detail, including the accuracy of FMCW, dead-reckoning, contour construction, and the impact of different factors on the performance of SAMS. Next, we compare with the *BatMapper* system [64].

5.1 FMCW Based Distance Measurement

Our chirp duration is 30 ms. It measures distances up to 5 m one-way. To quantify the FMCW based distance measurement accuracy, we put the smartphone on the floor with its speaker facing towards the wall at different distances and take 30 samples for each location. Fig. 15(a) shows the distance measurement is accurate across different distances. The distance error is stable as the distance increases. Even at 5 meter away, the FMCW based distance measurement error is within 2.5 cm. We have also repeated this experiment in the an-echoic chamber, and got similar results. Next, we plot the overall CDF of distance measurement errors across different distance in Fig. 15(b). As we can see, the median error is around 1.5 cm, which is good enough for contour mapping.



(a) FMCW Distance Measurement Error for Different Distances.
(b) CDF for FMCW based Distance Measurement Error.
(c) CDF of Step Size Estimation Error.
(d) Dead-reckoning Based Distance Estimation Error.

Fig. 15. Evaluation of FMCW Based Distance Measurement and Dead-reckoning Modules.

5.2 Dead-reckoning Based Estimation

As discussed earlier, in SAMS system, we have used a calibration-free step size estimation algorithm. We have evaluated this step size estimation algorithm across 3 users. We ask the users to walk 10 m distance for 20 times and record their numbers of steps to get the median step-size distance. Then, we take the difference from the median size with our calculated size. As shown in Fig. 15(c), the median error is less than 6 cm, which is reasonably good considering no personalized training is required. For these distances traveled, we have also analyzed the distance estimation, (which is a function of step size and step count). Fig. 15(d) shows CDF of distance calculation error: it is mostly within 1m across different movement speeds.

We first estimate the accuracy of our system in a setup created by wooden walls and then move on to map real rooms and corridors. Below we evaluate each of the sub-modules of contour construction.

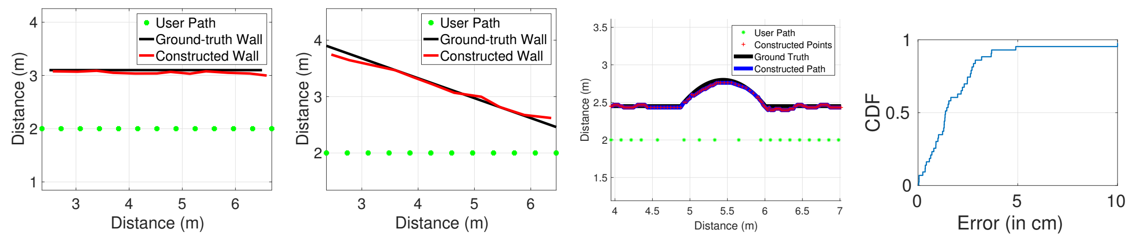
Table 2. Wall Association Accuracy and Outlier ratio.

Experiment	**Association Accuracy (%)**	**Outlier Ratio (%)**
Single Wall	100	0
Two Wall	90.4	6.4
Full Contour	88.1	9.6

5.3 Wall Association Accuracy

Table 2 summarizes the wall association accuracy and outlier ratio for different types of contours that we have evaluated. For full room contour, the accuracy is around 88%. The association error occurs mostly at the obtuse angled corners because the change in gradient is not very large. However, we have observed that wrong wall association at corner does not significantly affect our contour estimation since assigning corner points to either wall does not significantly change the gradient estimate or spline smoothing of either wall. For contours with four walls, the difference in median error from the ground truth is less than 1 cm between the ideal wall mapping and our wall assignment. The outliers also occur mostly at the corners, since multiple peaks at the corners merge to give a wider peak, moving the maximum point from its original place and hence giving us erroneous distance values, which results in outliers. Table 2 shows that increasing the number of corners increases the outlier ratio.

Next, we evaluate our combined SAMS system for recreating an indoor space. As mentioned earlier, initially, we have used wooden walls to create different shapes and used laser based system to get the ground-truth. To observe the capability of the system, we have evaluated the system incrementally: starting from a single wall, then to a wall intersection, and finally to a full contour construction, which uses both user path and the distance measurement to the nearby objects. We construct a room contour in the presence of clutters (*e.g.*, a small chair or desk). Finally, we evaluate SAMS in real rooms and corridors. Finally, we evaluate SAMS in real rooms and corridors. To quantify the *contour construction error*, for each point on the estimated contour, we compute the difference between the point and the closest point on the ground truth contour.



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

Fig. 16. (a) Straight Single Wall. (b) Angled Single Wall. (c) Curved Wall. (d) CDF of Single Wall Estimation Error.

5.4 Single Wall Contour

Straight Wall Contour. We create a single straight wall or angled wall through wooden structures. In these experiments, the user moves in a straight path holding the phone in hand with the phone speaker facing the wall. Fig. 16 shows the user path, ground-truth, and constructed path. All the experiments are performed 10 times, and the constructed contour with the median error has been plotted, unless specified otherwise. As we can see, the constructed shape is close to the ground-truth. As Fig. 16(d) shows, the median distance estimation error is within 4cm, and the measured angle between the user path and wall is within 1.5 degree.

Curved Wall Contour. We also test with wooden walls having curved surfaces. We observe that the system is able to correctly identify the presence of a curved surface in the contour and use spline interpolation [25] for contour construction. The CDF of error for curved wall contours is shown in Fig. 16(d). We observe the reconstructed contour is within 5cm from the ground truth. Fig. 16(c) shows an example of reconstructed curved wall contour.

5.5 Multi Wall Contour

To test SAMS for arbitrary shaped contour, we use wooden walls to construct different multi-wall shapes. First, we start with two-wall intersection having different angles. Fig. 17 (a) and (b) show that we can create the shape of the two wall intersection very accurately, irrespective of the intersection angles or user path patterns. Note that we have done linear regression on the constructed points after removing the outliers (due to junctions and sway movement) and corner points to generate the constructed path. Fig. 18(a) shows that for a two wall contour estimation the median error is less than 10 cm. We note that rooms with an arbitrary number of walls can be constructed using different combinations of two-wall contours.

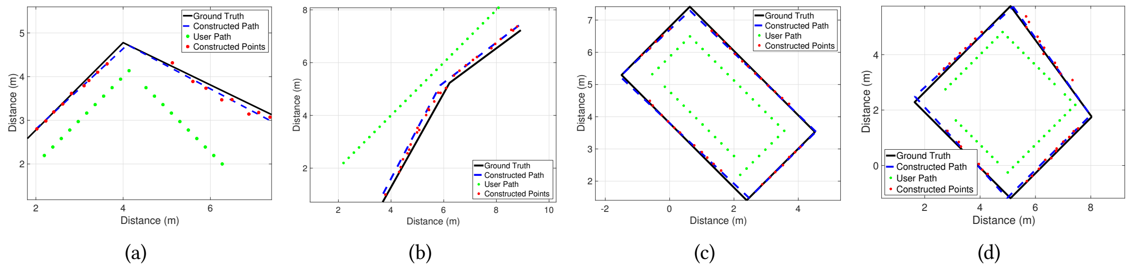


Fig. 17. (a) 120 Degree Corner. (b) 230 Degree Corner. (c) Rectangular Room. (d) Trapezoid Room.

Finally, we move on to create closed multi-wall room shaped contour. Fig. 17 (d) and (e) illustrate that our system can create full room contours accurately in a single step. We use four-wall enclosures as examples to show the applicability of our system.

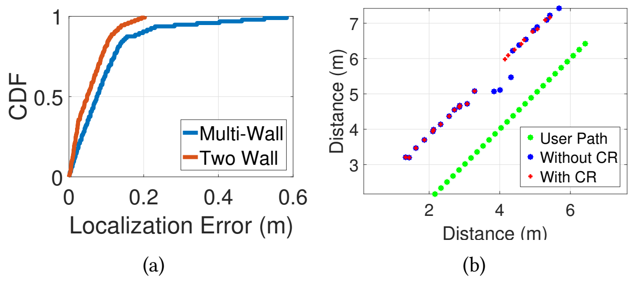


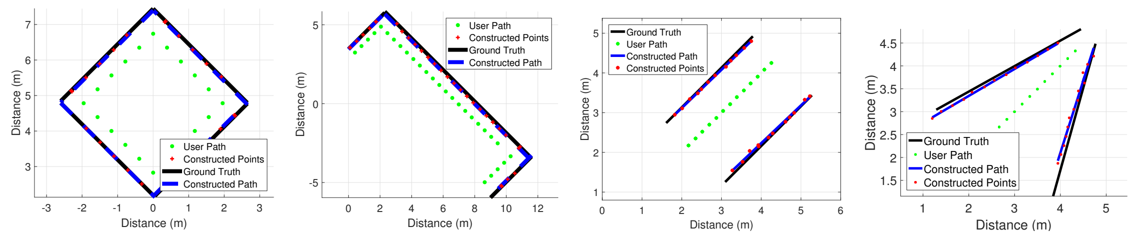
Fig. 18. (a) CDF of Distance Estimation Error. (b) Clutter Removal in Action.

5.6 Contour with Clutter

We define static clutters as the *smaller* things that block the direct path of audio from smartphone to the wall. By *smaller* things, we mean the width of the object is within 60cm and the height would be such that it will block the direct path. To evaluate the impact of static clutter in room contour formation, we put a plastic chair (of the desired dimension) and small wooden cabinet in front of a straight wall and the user walked across the wall in a straight path. As shown in Fig. 18(b), the chair in front of the wall generates a set of points not in-line with the majority of the points of the straight wall. So, the outlier removal will automatically remove the clutter. For bigger clutters like couches, our algorithm will identify it as a separate object. This is reasonable since a large object can be considered as part of the room contour. We leave to the future work to further distinguish different types of clutters.

5.7 Creating Maps of Rooms and Corridors

We use SAMS to map real room and corridor contours in the presence of furniture in our building. The ability to map arbitrary shaped rooms has already been evaluated in section 5.5. Fig. 19(a) and Fig. 19(b) further illustrate that SAMS can map real rooms and corridor in a single step with a median error of 12 cm.



(a) Rectangular Room Mapping. (b) Corridor Mapping. (c) Parallel Corridor Mapping. (d) Non-parallel Corridor Mapping.

Fig. 19. Mapping the Layouts of Real Rooms and Corridors.

Next, we also explore the possibility of using two mics to further reduce the user profiling effort. Our experiments show that it is difficult to get reliable FMCW based distance information from the top mic if the speaker is at the bottom (*e.g.*, in Samsung Galaxy S7). Therefore, we use Samsung Galaxy S4, whose speaker is at the rear side and microphones are at the two ends, so that the received signals at both microphones are strong. User holds the phone in such a way that the mics face the opposite walls. These two simultaneous distances captured via both mics are used to map the walls of the corridor as shown in Fig. 19(c) and the median error is around 8cm.

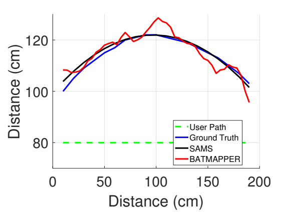
5.8 Comparison with *BatMapper* System

In this section, we compare the results with echo-driven *BatMapper* [64] with ours in the following settings: distance estimates from a static reflector, reconstruction of curved surfaces, reconstruction of a room, and mapping of an indoor floor in a building. *BatMapper* employs acoustic echo correlation based distance estimation technique that was previously proposed in [29]. However, its accuracy is low due to its less accurate distance estimation than our customized FMCW and the lack of general contour construction algorithm. Furthermore, it requires initial training for estimating parameters for the underlying algorithm, multiple walks to get better results, and has limited range (up to 3.5m one way).

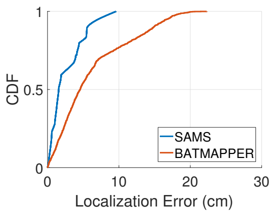
First, we compare the distance estimation for static reflectors (from 1m to 5m with an incremental increase of 0.5 m) between our FMCW and their echo and find SAMS yields 0.5 cm to 2.5 cm median distance error, whereas *BatMapper* yields 1 cm to 15 cm median error. Next we consider contour construction and find our scheme can construct curved surfaces much better (4 cm median contour error and 8 cm 90th percentile error in our algorithm versus 6 cm median error and 18 cm 90-percentile error in *BatMapper*) as shown in Fig. 20(b). Since we consider geometric constraints carefully, our curved surface formation is clearly better as shown in Fig. 20(a). The worst case error of *BatMapper* system can go up to 22 cm, as shown in Fig. 20(b).

We further evaluate room construction (Fig. 20(c)) and observe that the median error of *BatMapper* is 8 cm more than SAMS. In the extreme cases, which often occur in curved or irregular shapes, the difference in the estimation error can go up to 12 cm (Fig. 20(d)).

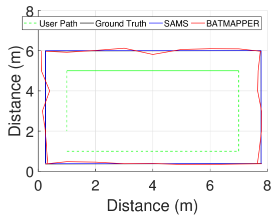
Next, we evaluate floor plan construction. In this experiment, the user maps each segment separately. To enable SAMS to stitch these different contours together to produce a complete floor-map, SAMS requires explicit user input. While walking a segment or a closed loop, the user presses a button in the app to record the start and end points, which instructs the phone to record the IMU sensors and time-stamps at those points. When the user starts a new walking loop, it picks one of the recorded points on a previous walking loop as the starting point, and another one as the ending point for that loop. The app will automatically stitch multiple section contours together to produce a complete floor-map. Fig. 21(c) illustrates that the indoor map contour estimation error can go up to 2.6 m for *BatMapper*, whereas the error of our system is 1.2 m. Note that stitch is not required in general, and a user can also construct a floor plan by walking along the floor in one shot. Stitching is just an option when the floor is too large to walk in one run.



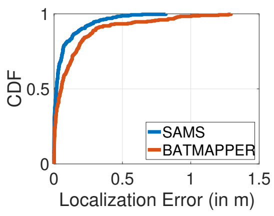
(a) Example of Curved Surface Estimation.



(b) CDF Error for Curve Surface Estimation.

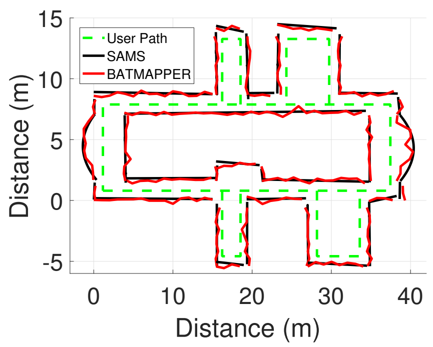


(c) Example of Room Contour Estimation.

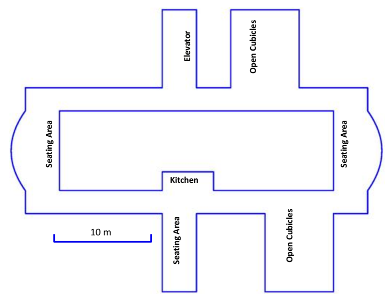


(d) CDF Error for Room Surface Estimation.

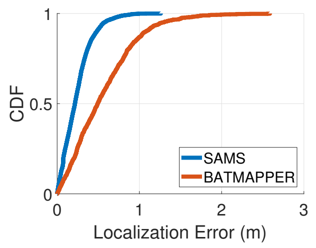
Fig. 20. Comparison between *BatMapper* and SAMS to Create Layouts.



(a) A Floor Space Mapping.



(b) Actual Floor Space Map.



(c) CDF of Indoor Floor Map Estimation Error.

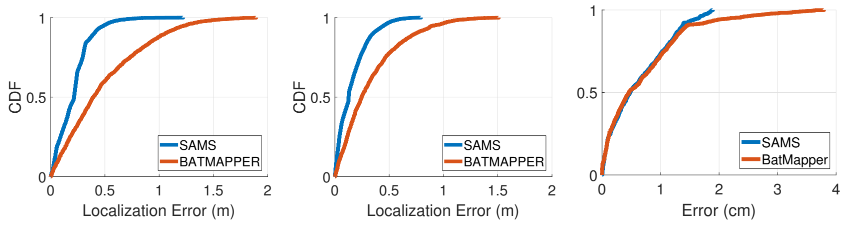
Fig. 21. Mapping of a Real Floor Space using SAMS and *BatMapper*.

5.8.1 Component Level Comparison. We further quantify the benefit of each component in our design.

Distance Measurement. SAMS uses FMCW to estimate the distance whereas *BatMapper* uses pulse based ranging. We compare these two schemes by varying the distances to a reflector from 0.4 m to 3 m. We have conducted these experiments in different indoor settings to compare performance in several multi-path scenarios. As seen in Fig. 22(c), the accuracy for distance estimation is similar for both schemes except for the tail above 90th percentile where SAMS is better than *BatMapper*. The tail in the CDF for *BatMapper* is caused by the error in measurements for distances less than 50 cm. This is because for these distances the reflected pulse overlaps with the transmitted *BatMapper*'s signal, making the reflected peak very difficult to detect.

Dead Reckoning. SAMS's dead reckoning component builds on the same prior work, with some minor modifications, used by *BatMapper* so the dead reckoning error is the same in both cases.

Contour Construction. In *BatMapper* system the estimates for wall distance based on its probabilistic peak selection are combined with user position estimates from dead reckoning to construct contours. *BatMapper* does not consider geometric constraints that are present due to the corridor and the room shapes; SAMS adds these geometric constraints on top of the constructed points to get a better estimate of contour and increase the accuracy of constructed contour. In Fig. 22, we compare accuracy by feeding the same distance and dead reckoning path estimates to both scheme's contour construction module. Fig. 22(a) illustrates the scenario when there is a curved surface present and in Fig. 22(b) we only have straight walls. It can be seen from the figures that the majority of the performance benefit of SAMS over *BatMapper* comes from this module.



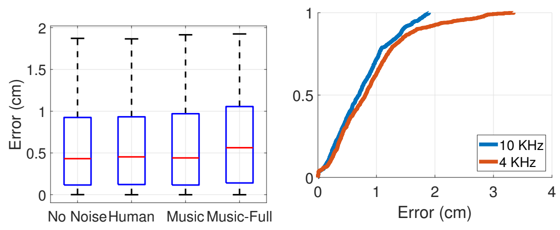
(a) Contour Construction Error with Curved Surfaces. (b) Contour Construction Error with Straight Surfaces Only. (c) Distance Estimation Error.

Fig. 22. Comparing SAMS and *BatMapper*.

5.9 Impact of Different Factors on SAMS

In this sub-section, we evaluate the robustness of our system against a few important factors.

Robustness to Ambient Sound. We note that all the results shown are done in the presence of common noise in normal office environment, including noise from HVAC system and people. Next we have further stress-tested our system by continuously talking and/or playing music at normal and loud volumes. We have played several different genres of music (*e.g.*, Jazz, Pop etc.) together to emulate different ambient sound. The noise source is (played through a logitech speaker) placed 10 cm away from the smartphone that is running SAMS. Fig. 23(a) compares the distance estimation error in normal office environment for different noise sources. We observe that human voice and medium volume music have negligible effect; playing music at the highest volume changes the median error by less than 0.5 cm. This indicates that our scheme is robust to ambient sound.



(a) Distance Estimation Accuracy in Presence of Noise. (b) Impact of Bandwidth on Distance Estimation.

Fig. 23. Impact of Different Factors on SAMS.

Effect of Doppler Shift. In SAMS's use-case scenarios, we always measure the perpendicular distance from the wall. This means that the *relative radial velocity* between in direction of the reflected signal is zero as the walking direction is perpendicular to the reflected path. Therefore, the reflected peak in FMCW spectrum is unaffected from Doppler effect. To verify this observation, we have measured the frequency shift in the reflected signal of a pure tone. We have gathered 30 traces of user moving along the wall in different indoor settings: corridor, lab, office, while keeping the phone's speaker facing towards the wall. Therefore, even though we are moving with a higher speed, the relative radial velocity between the smartphone and the wall is zero as the walking direction is perpendicular to the direction of reflected wave, thus the Doppler shift for the reflected wave should be close to zero. We observed that for a normal walking speed (*e.g.*, 1m/s), the maximum Doppler shift is within 1 Hz [2]. This small shift seen in our experiment is due to the user's hand movement, which has non-zero radial velocity (3 cm/s). However, this translates into a distance estimation error of less than a millimeter. Therefore, for all practical purposes, the effect of Doppler shift on SAMS is negligible.

Impact of Bandwidth. SAMS uses a sweep frequency band of 11 KHz to 21 KHz. As shown in Fig. 5(b), this bandwidth is required to distinguish multiple nearby reflectors. However, this audio signal is audible. We use 30% volume level in our experiment. This level is barely audible to human ear. Moreover, indoor floor-map construction is usually one-time and small audible sound is acceptable.

To further test the impact of sweep frequency bandwidth, we further evaluated distance estimation accuracy for the signal between 18 to 22 KHz. We compared the error in distance estimation for both bandwidths while measuring distance to the walls in a corridor. Results in Fig. 23(b) show that the difference in accuracy is less than 0.2 cm for the 80% of the values. However, we do see some large errors for around 20% of the measurements. This is because for most of the measurements the only reflector is the wall so we get only a single peak in the FMCW profile. However, in the presence of some furniture close to the wall, we get multiple reflection. Smaller bandwidth leads to merged peaks, which may shift the peak and results in a larger error. These results indicate reducing the bandwidth to 18 to 22 KHz results in similar accuracy in most cases except in presence of clutter, in which case higher errors may arise.

6 USE CASE

As a use case scenario, we show that the constructed indoor contour can be used to predict the wireless signal strength. We develop a simple ray-tracer to estimate Wi-Fi received signal strength (RSS) at different locations using the indoor contour constructed from SAMS. The RSS is estimated as

$$RSS = P_t + G_t + G_r - (\alpha * 10\log(d) + R_l)$$

where P_t is the transmitted power, G_t and G_r are the TX and RX gain, respectively,
alpha is the attenuation factor and R_l is the reflection loss from the wall. In our experiments, we use a laptop with Intel 5300 chip acting as a transmitter and another laptop acting as a receiver logs the RSS for every packet using Intel CSI toolkit [37]. The transmission power (P_t) and antenna gains (G_t and G_r) remain the same during all our experiments and can be estimated jointly. To estimate these parameters, we measure RSS at different distances from TX in an open space so that there is only a direct path. Using these measurements, we estimate $P_t + G_t + G_r$ and α. We then perform another experiment with TX and RX placed in front of a single wall. Using previous estimates and RSS readings at different distances from the single wall, we estimate R_l. The wall material is same for all the experiments done later, so these estimated parameters remain same for all the experiments. Now we use simple ray-tracing to estimate the received signal of the direct path and all first order reflected paths from the transmitter to the receiver given a room layout. Using the TX power, attenuation factor, and reflection loss estimates, the ray tracer synthesizes signals from all paths and outputs the RSS at the RX location. The room layout generated using SAMS is given to the ray-tracing engine. It is used to predict RSS by calculating possible paths at different distances in different settings. We compare the estimated RSS with the measured RSS by placing TX and RX in a corridor of length 10 m, in a closed rectangular room of dimensions 6 m x 4 m and in front of a

[2]Using Doppler equation $v = \delta f * c/f_0$, This corresponds to velocity of 3 cm/s, for center frequency of 11 KHz

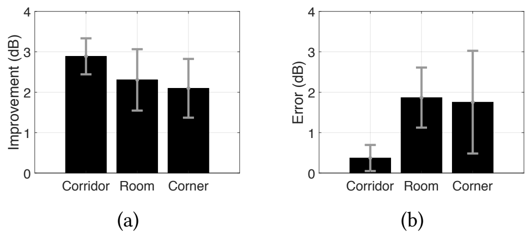


Fig. 24. (a) Improvement Over Baseline Model. (b) Difference Between Predicted and Measured RSS.

corner in a large hall. In all three settings, we fix the position of transmitter and measure the RSS at receiver at 10 different points with varying separation between TX and RX and the compare them with SAMS ray-tracer. Fig. 24 shows that SAMS ray-tracer is able to improve the RSS prediction by up to 3 dB over the baseline, which only considers the direct path. Moreover, the predicted RSS is within 1.5 to 2 dB of the measured values. These results suggest we can use the contour to predict the RSS at any given location (even without any direct measurements at that location). Such information can be used for a number of important wireless optimization tasks, such as AP placement, AP selection, and rate adaptation.

7 RELATED WORK

Recently, there is an increased interest in the research community in acoustic based applications. In the following, we present those works which are related in terms of techniques or applications.

Acoustic Tracking: For device-based acoustic tracking, AAMouse [62] uses doppler shift based acoustic tracking, whereas CAT [45] leverages external speakers and uses FMCW for phone movement tracking at mm-level accuracy. Device-free techniques like LLAP [61] utilizes phase-shift, and FingerIO [48] leverages echo-profile correlation, to achieve 1 cm accuracy. In our application setting, phase-based and echo-profile do not perform well. Our SAMS is built on top of FMCW-based distance. Different from existing work, our goal is to go beyond distance measurement to construct an indoor map.

Acoustic Ranging: Graham et al. [29] use smartphones to show echo peaks to create a software-based sonar sensor for distance measurements. [36] further builds a sonar attachment for smartphone to measure distance and [60] uses correlation based FMCW to detect breathing movement within a small range. DopEnc [63] uses an acoustic-based approach for identifying persons that the user encounters during interaction. Recently, researchers have combined depth sensor, camera, and audio to create a LiDAR like attachment for smartphones [33]. Different from these works, our system does not require extra hardware while achieving high accuracy.

Comparison with Depth Cameras: There are four main approaches to gather depth information from a single scene or image: i) stereo vision based approach, ii) structured light based methodology, iii) time-of-flight based approach, and iv) data-driven learning based approach. For example, Bumblebee camera [7] uses stereo vision based approaches, which requires proper calibration with a host pc and parameter tuning for new environments to get accurate measurements. In the case of smartphone based acoustic sensing, the ubiquity of speaker and microphone help in adoption. In comparison, Kinect uses structured light based technology, which works up to 3 m range accurately but does not provide good results after that. The range of our system can go up to 5 m. The PMD CamCube [16], like LiDAR, a time-of-flight (ToF) camera, acquires depth information directly on the camera hardware without any calibration to provide good depth estimates. This camera has several limitations: high cost, noisy depth estimation, sensitive to ambient light or transparent glass like material, which limits its use to indoor scenarios with controlled lighting. Acoustic based approach can work in a variety of settings, both for transparent material and in dark situation. Furthermore, issues related to image capture like orientation or occlusion will not affect our system. Recently, deep-learning based methods are used to estimate the depth information of different objects in a particular image [35]. This can be achieved by training from a manually labeled data of background/fore-ground images and objects combined. Recently, Google Pixel smart-phone enables *portrait* mode in a single camera based setting [17], by partially utilizing this method. However, it is tailored to a specific set of problems and requires re-training for each of the cases in different scenarios.

Indoor Mapping: Indoor mapping and floor plan construction have become an urgent problem for location based applications. Robotic approaches can produce accurate maps, but they usually require expensive special hardware (*e.g.*, laser rangers [57], depth cameras [40], sonars [58]). Vision based techniques [18, 55] can generate 3*D* models of building interiors, but they incur high computing overhead, and also face privacy and technical limitations (*e.g.*, glass walls, blurry images). Recently, CrowdInside [22] uses inertial data with anchor points to approximate shapes of accessible areas through crowd-sourcing. Jiang et al. [39] leverage Wi-Fi signatures to detect room and hallway adjacency, and combine with user trajectories to construct hallways. Jigsaw [34] and other works [27, 31] leverage images to generate geometry attributes and spatial constraints of indoor landmarks. Such work usually requires significant amount of data and crowd-sourcing efforts due to limited resources and energy on the smartphone. Moreover, those using images/videos also face privacy restrictions.

Due to the high computation overhead, high energy cost, and privacy concern of vision-based approaches, researchers have attempted to build acoustic mapping solutions. Recently, researchers were able to create a convex polygon-shaped room with the help of multiple mics positioned at known locations [30, 42, 52]. They utilize echoes and geometric property of echo-arrival in a convex polygonal room to construct the room-shape, based on the known locations of mics. *BatMapper* is the closest to our work. It is an *infrastructure-free* smartphone based acoustic mapping system. The main differences are the following: (i) it requires extensive training to infer the parameters of the probabilistic algorithm, (ii) it uses echo based distance estimation, which has higher error, (iii) it cannot handle surfaces with general shapes, and (iv) it requires multiple trials to construct the map of a single space. Compared with the existing works, the advantages of our approach is that we can construct a indoor map for surfaces with arbitrary shapes using a smartphone without reliance on any infrastructure or calibration by simply letting a user walk around the area *once*. It removes the need of infrastructure deployment for higher accuracy and the need of extensive training step.

8 CONCLUSION

In this paper, we present a smartphone-based indoor space mapping system. It first measures the distance to one or more nearby objects. Then, it uses customized IMU-based dead-reckoning and geometric constraints derived from the measured distances to construct the room contour. Our evaluation results show the promise of our system. As part of our future work, we plan to explore other specific applications of SAMS, such as indoor localization, navigation, and enriching AR/VR.

REFERENCES

[1] 2017. Affordable Scanning LiDAR for everyone. http://scanse.io. [Online] Last Accessed : 11/30/2017.
[2] 2017. AirLoc: Pedestrian dead reckoning for passenger localization. http://www.es.ewi.tudelft.nl/msc-theses/2017-deMoes.pdf. [Online] Last Accessed : 11/30/2017.
[3] 2017. Android ARCore. https://developers.google.com/ar/. [Online] Last Accessed : 11/30/2017.
[4] 2017. Apple ARKit. https://developer.apple.com/arkit/. [Online] Last Accessed : 11/30/2017.
[5] 2017. AR MeasureKit. https://itunes.apple.com/app/id1258270451. [Online] Last Accessed : 11/30/2017.
[6] 2017. Background noise reduction: one of your smartphone's greatest tools. http://www.techradar.com/news/phone-and-communications/mobile-phones/background-noise-reduction-one-of-your-smartphone-s-greatest-tools-1229667. [Online] Last Accessed : 11/30/2017.
[7] 2017. Bumblebee Camera. https://www.ptgrey.com/bumblebee2-firewire-stereo-vision-camera-systems. [Online] Last Accessed : 30/11/2017.

[8] 2017. Cook's Distance. http://bit.ly/2n2iQtv. [Online] Last Accessed : 11/30/2017.
[9] 2017. Housecraft App. https://itunes.apple.com/us/app/housecraft/id1261483849. [Online] Last Accessed : 11/30/2017.
[10] 2017. IKEA Place App. https://itunes.apple.com/us/app/ikea-place/id1279244498. [Online] Last Accessed : 11/30/2017.
[11] 2017. IPhone X. www.apple.com/iphone-x/. [Online] Last Accessed : 11/30/2017.
[12] 2017. Leica DISTO E7100i 200ft Laser Distance Measure with Bluetooth. http://amzn.to/2m2WSr4. [Online] Last Accessed : 11/30/2017.
[13] 2017. Lenovo Phab Series Smartphones. https://www3.lenovo.com/us/en/smart-devices/-lenovo-smartphones/phab-series/c/phab-series. [Online] Last Accessed : 11/30/2017.
[14] 2017. Microsoft Hololens. https://www.microsoft.com/microsoft-hololens/en-us. [Online] Last Accessed : 11/30/2017.
[15] 2017. Oculus VR Rift headset. https://www.oculus.com/rift/. [Online] Last Accessed : 11/30/2017.
[16] 2017. PMD[Vision] Camcube Depth Camera. https://www.youtube.com/watch?v=f0bYnxy74fE. [Online] Last Accessed : 30/11/2017.
[17] 2017. Portrait mode on the Pixel 2 and Pixel 2 XL smartphones. https://research.googleblog.com/2017/10/portrait-mode-on-pixel-2-and-pixel-2-xl.html. [Online] Last Accessed : 30/11/2017.
[18] 2017. Project Tango. https://get.google.com/tango/. [Online] Last Accessed : 11/30/2017.
[19] 2017. Sample compass code. https://github.com/iutinvg/compass. [Online] Last Accessed : 11/30/2017.
[20] 2017. THe average walking stride length. http://www.livestrong.com/article/206902-how-to-reset-a-new-balance-sport-pedometer/. [Online] Last Accessed : 11/30/2017.
[21] Moustafa Alzantot and Moustafa Youssef. 2012. CrowdInside: Automatic Construction of Indoor Floorplans. In *SIGSPATIAL '12*. ACM, New York, NY, USA, 99–108.
[22] Moustafa Alzantot and Moustafa Youssef. 2012. CrowdInside: Automatic Construction of Indoor Floorplans. In *SIGSPATIAL '12*. ACM, New York, NY, USA, 99–108.
[23] S. Ayub, B. M. Heravi, A. Bahraminasab, and B. Honary. 2012. Pedestrian Direction of Movement Determination Using Smartphone. In *2012 Sixth International Conference on Next Generation Mobile Applications, Services and Technologies*. 64–69.
[24] Martin Azizyan, Ionut Constandache, and Romit Roy Choudhury. 2009. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *MobiCom '09*. 261–272.
[25] Richard H. Bartels, John C. Beatty, and Brian A. Barsky. 1987. *An Introduction to Splines for Use in Computer Graphics &Amp; Geometric Modeling*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
[26] Si Chen, Muyuan Li, Kui Ren, and Chunming Qiao. 2015. Crowd Map: Accurate Reconstruction of Indoor Floor Plans from Crowdsourced Sensor-Rich Videos. In *ICDCS 2015, Columbus, OH, USA*. 1–10.
[27] S. Chen, M. Li, K. Ren, and C. Qiao. 2015. Crowd Map: Accurate Reconstruction of Indoor Floor Plans from Crowdsourced Sensor-Rich Videos. In *2015 IEEE 35th International Conference on Distributed Computing Systems*. 1–10.
[28] R. Dennis Cook. 1979. Influential observations in linear regression. *J. Amer. Statist. Assoc.* 74, 365 (1979), 169–174.
[29] Graham Daniel, Simmons George, T.Nguyen David, and Zhou Gang. 2015. A Software Based Sonar Ranging Sensor for Smart Phones. *IoT* (2015).
[30] Ivan Dokmanića, Reza Parhizkara, Andreas Walthera, Yue M. Lub, and Martin Vetterlia. 2013. Acoustic echoes reveal room shape. *Proceedings of National Academy of Sciences (PNAS)* (2013).
[31] Jiang Dong, Yu Xiao, Marius Noreikis, Zhonghong Ou, and Antti Ylä-Jääski. 2015. iMoon: Using Smartphones for Image-based Indoor Navigation. In *SenSys*. ACM, 85–97.
[32] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. 2015. Visual Simultaneous Localization and Mapping: A Survey. *Artif. Intell. Rev.* 43, 1 (2015), 55–81.
[33] J. H. Gao and Li-Shiuan Peh. 2016. A smartphone-based laser distance sensor for outdoor environments. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2922–2929.
[34] Ruipeng Gao, Mingmin Zhao, Tao Ye, Fan Ye, Yizhou Wang, Kaigui Bian, Tao Wang, and Xiaoming Li. 2014. Jigsaw: Indoor Floor Plan Reconstruction via Mobile Crowdsensing. In *MobiCom '14*. ACM, 249–260.
[35] Ravi Garg, Vijay Kumar B. G, and Ian D. Reid. 2016. Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue. *CoRR* abs/1603.04992 (2016). arXiv:1603.04992 http://arxiv.org/abs/1603.04992
[36] Daniel Graham, Gang Zhou, Ed Novak, and Jeffrey Buffkin. 2016. A Smartphone Compatible SONAR Ranging Attachment for 2-D Mapping. *IoT* (2016).
[37] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool Release: Gathering 802.11n Traces with Channel State Information. *ACM SIGCOMM CCR* 41, 1 (Jan. 2011), 53.
[38] Yifei Jiang, Yun Xiang, Xin Pan, Kun Li, Qin Lv, Robert P. Dick, Li Shang, and Michael Hannigan. 2013. Hallway Based Automatic Indoor Floorplan Construction Using Room Fingerprints. In *UbiComp '13*. ACM, 315–324.
[39] Yifei Jiang, Yun Xiang, Xin Pan, Kun Li, Qin Lv, Robert P. Dick, Li Shang, and Michael Hannigan. 2013. Hallway Based Automatic Indoor Floorplan Construction Using Room Fingerprints. In *UbiComp '13*. ACM, New York, NY, USA, 315–324.
[40] Kourosh Khoshelham and Er Oude Elberink. 2012. Accuracy and resolution of kinect depth data for indoor mapping applications. In *Sensors, 12, 1437–1454. 2013*. 8238.

[41] Kourosh Khoshelham and Er Oude Elberink. 2013. Accuracy and resolution of kinect depth data for indoor mapping applications. In *Sensors 2012, 12, 1437–1454*. 8238.

[42] Miranda Krekovic, Ivan Dokmanic, and Martin Vetterli. 2016. EchoSLAM: Simultaneous localization and mapping with acoustic echoes. In *ICASSP 2016, Shanghai, China, March 20-25, 2016*. IEEE, 11–15.

[43] Lawrence Berkeley National Laboratory, United States. Department of Energy. Office of Scientific, and Technical Information. 2001. *The National Human Activity Pattern Survey (NHAPS): A Resource for Assessing Exposure to Environmental Pollutants*. Lawrence Berkeley National Laboratory. https://books.google.com/books?id=njh3AQAACAAJ

[44] Kaikai Liu, Xinxin Liu, and Xiaolin Li. 2013. Guoguo: Enabling Fine-grained Indoor Localization via Smartphone. In *MobiSys '13*. 235–248.

[45] Wenguang Mao, Jian He, and Lili Qiu. 2016. High-precision Acoustic Motion Tracking. In *MobiCom '16*. 491–492.

[46] Yuya Murata, Katsuhiko Kaji, Kei Hiroi, and Nobuo Kawaguchi. 2014. Pedestrian Dead Reckoning Based on Human Activity Sensing Knowledge. In *UbiComp '14 Adjunct*. ACM, New York, NY, USA, 797–806.

[47] Rajalakshmi Nandakumar, Shyamnath Gollakota, and Nathaniel Watson. 2015. Contactless Sleep Apnea Detection on Smartphones. In *MobiSys '15*. ACM, New York, NY, USA, 45–57.

[48] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota. 2016. FingerIO: Using Active Sonar for Fine-Grained Finger Tracking. In *CHI '16*. 1515–1525.

[49] Yasuharu Onishi, Jun Kuroda, Yukio Murata, Motoyoshi Komoda, Kazuyuki Tsunoda, Yukio Yokoyama, Yasuhiro Oikawa, and Yoshio Yamasaki. 2010. *The acoustical design of mobile phones*. Vol. 2. 1250–1257.

[50] Chunyi Peng, Guobin Shen, Yongguang Zhang, Yanlin Li, and Kun Tan. 2007. BeepBeep: A High Accuracy Acoustic Ranging System Using COTS Mobile Devices. In *SenSys '07*. ACM, New York, NY, USA, 1–14.

[51] Nirupam Roy, He Wang, and Romit Roy Choudhury. 2014. I am a smartphone and I can tell my users walking direction. In *In Proceedings of ACM International Conference onMobile Systems, Applications, and Services (MobiSys)*.

[52] R. Scheibler, I. Dokmanić, and M. Vetterli. 2015. Raking echoes in the time domain. In *ICASSP '15*. 554–558.

[53] Wen-Yuah Shih, Liang-Yu Chen, and Kun-Chan Lan. 2012. Estimating Walking Distance with a Smart Phone. In *PAAP '12*. IEEE Computer Society, Washington, DC, USA, 166–171.

[54] Yuanchao Shu, Kang G. Shin, Tian He, and Jiming Chen. 2015. Last-Mile Navigation Using Smartphones. In *MobiCom '15*. ACM, New York, NY, USA, 512–524.

[55] Keith N. Snavely. 2009. *Scene Reconstruction and Visualization from Internet Photo Collections*. Ph.D. Dissertation. Seattle, WA, USA. Advisor(s) Seitz, Steve and Szeliski, Rick.

[56] Arno Solin, Santiago Cortes, Esa Rahtu, and Juho Kannala. 2017. Inertial Odometry on Handheld Smartphones. *CoRR* abs/1703.00154 (2017).

[57] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. 2003. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Robotics and Autonomous Systems* 3-4 (2003), 181–198.

[58] Juan D. Tardós, José Neira, Paul M. Newman, and John J. Leonard. 2002. Robust Mapping and Localization in Indoor Environments Using Sonar Data. *I. J. Robotic Res.* 21, 4 (2002), 311–330.

[59] He Wang, Souvik Sen, Ahmed Elgohary, Moustafa Farid, Moustafa Youssef, and Romit Roy Choudhury. 2012. No Need to War-drive: Unsupervised Indoor Localization. In *MobiSys '12*. ACM, New York, NY, USA, 197–210.

[60] Tianben Wang, Daqing Zhang, Yuanqing Zheng, Tao Gu, Xingshe Zhou, and Bernadette Dorizzi. 2017. C-FMCW Based Contactless Respiration Detection Using Acoustic Signal. *IMWUT* 1, 4 (2017), 170:1–170:20.

[61] Wei Wang, Alex X. Liu, and Ke Sun. 2016. Device-free Gesture Tracking Using Acoustic Signals. In *MobiCom '16*. ACM, New York, NY, USA, 82–94.

[62] Sangki Yun, Yi-Chao Chen, and Lili Qiu. 2015. Turning a Mobile Device into a Mouse in the Air. In *MobiSys '15*. 15–29.

[63] Huanle Zhang, Wan Du, Pengfei Zhou, Mo Li, and Prasant Mohapatra. 2016. DopEnc: Acoustic-based Encounter Profiling Using Smartphones. In *MobiCom '16*. 294–307.

[64] Bing Zhou, Mohammed Elbadry, Ruipeng Gao, and Fan Ye. 2017. BatMapper: Acoustic Sensing Based Indoor Floor Plan Construction Using Smartphones. In *MobiSys '17*. ACM, New York, NY, USA, 42–55.

[65] Pengfei Zhou, Mo Li, and Guobin Shen. 2014. Use It Free: Instantly Knowing Your Phone Attitude. In *MobiCom '14*. ACM, 605–616.

Received November 2017; revised February 2018; accepted April 2018