Centaur: Locating Devices in an Office Environment

Rajalakshmi Nandakumar, Krishna Kant Chintalapudi, Venkata N. Padmanabhan
Microsoft Research India

ABSTRACT

We consider the problem of locating devices such as laptops, desktops, smartphones etc. within an office environment, without requiring any special hardware or infrastructure. We consider two widely-studied approaches to indoor localization: (a) those based on Radio Frequency (RF) measurements made by devices with WiFi or cellular interfaces, and (b) those based on Acoustic Ranging (AR) measurements made by devices equipped with a speaker and a microphone. A typical office environment today comprises devices that are amenable to either one or both these approaches to localization. In this paper we ask the question, *"How can we combine RF and AR based approaches in synergy to locate a wide range of devices, leveraging the benefits of both approaches?"* The key contribution of this paper is Centaur, a system that fuses RF and AR based localization techniques into a single systematic framework that is based on Bayesian inference. Centaur is agnostic to the specific RF or AR technique used, giving users the flexibility of choosing their preferred RF or AR schemes. We also make two additional contributions: making AR more robust in non-line-of-sight settings (EchoBeep) and adapting AR to localize speaker-only devices (DeafBeep). We evaluate the performance of our AR enhancements and that of the Centaur framework through microbenchmarks and deployment in an office environment.

Categories and Subject Descriptors

C.2.m [**Computer Systems Organization**]: COMPUTER - COMMUNICATION NETWORKS—*Miscellaneous*

Keywords

Indoor localization, WiFi, acoustic ranging, Bayesian inference

1. INTRODUCTION

Locating mobile devices such as laptops and smartphones in office environments, can enable several pervasive computing applications. Furthermore, even locating less portable devices such as desktop computers and printers can be useful for applications such as asset tracking.

MobiCom'12, August 22–26, 2012, Istanbul, Turkey.

A large body of research over the past two decades has been dedicated to enabling indoor localization. Much of this research has fallen into one of two categories based on the nature of measurements that participant devices are expected to perform: (a) *Radio Frequency* (RF)-based techniques that rely on measuring the strength of signals from proximate RF sources (such as WiFi access points or cell towers) [2, 8, 16, 5], and (b) *Acoustic Ranging* (AR)-based techniques that rely on range measurements between proximate devices [15, 12]. Devices with WiFi or cellular interfaces are amenable to RF-based localization while those with speakers and microphones can be localized using acoustic ranging.

A modern day office environment has devices with one or both of WiFi interfaces and speakers/microphones. Devices such as laptops, equipped with both WiFi interfaces, and speakers/microphones, are amenable to both approaches to localization. Devices such as desktop computers are typically equipped with speakers and can participate only in AR-based localization techniques, although the absence of a microphone in a desktop computer is a key challenge that needs to be overcome. Further, each of these approaches to localization is associated with its own set of constraints and error characteristics. RF-based localization techniques have a wide coverage, given the ubiquity of WiFi in indoor environments. However, the inherent variability in RF propagation characteristics limits the accuracy of such techniques to the range of a few meters to a few tens of meters. On the other hand, AR-based trilateration is typically very accurate and can enable localization to the sub-meter level. However, these techniques have limited coverage since these depend on a very high device deployment density, requiring three or more devices with known locations to be within audible range (typically 10m) to enable trilateration. *In this paper we ask the question: "How can we combine both these approaches in synergy and achieve the benefits of both?" The key contribution of this paper is* Centaur, *a systematic framework that fuses RF and AR based localization techniques, using Bayesian inference.*

The Centaur framework comprises three different categories of devices. First, there are *anchors*, which are fixed devices whose locations are known a priori (e.g., desktop PCs assigned to specific employees) and which at least have a speaker and so can be made to emit sound to aid in the localization of other devices. Second, there are *dual-mode devices*, which have WiFi interfaces, as well as speakers/microphones, and thus are amenable to both methods of localization (e.g., laptops, smartphones). Finally, there are *speaker-only devices*, which can only emit a sound in order to enable their localization (e.g., desktop computers whose locations are not known a priori).

WiFi measurements performed by dual mode devices help obtain an initial probability distribution of the possible locations of these devices. Acoustic measurements amongst dual-mode devices and

between these devices and anchors, provide geometric constraints (e.g. a distance constraint) that relate the locations of these devices. Centaur, then uses all available initial probability distributions of dual-mode devices and inter-device geometric constraints to simultaneously refine the initial probability distributions of dual-mode devices and compute the probability distribution of speaker-only devices.

Acoustic measurements provide two kinds of geometric constraints in Centaur namely, *Inter-device Distance Constraints* (IDC) and *Inter-device Distance Difference Constraints* (IDDC), and each of these presents interesting challenges. When two devices within audible proximity each has a microphone and a speaker (e.g., laptops and smartphones), techniques such as BeepBeep [11] can be used to estimate the distance between them, providing an IDC for each such pair. However, one significant challenge we faced in our deployment is that devices are seldom in direct line of sight because of walls, furniture, and clutter in an office environment. State-of-the-art acoustic ranging techniques do not perform well in such a setting because the direct line-of-sight acoustic signal could be much weaker than a delayed but stronger reflected signal. To address this problem, in Section 4 we propose an improved ranging scheme, *EchoBeep*, that performs significantly better than existing ranging schemes when the line-of-sight signal is much weaker than the reflected "echoes".

In contrast to laptops and smartphones, devices such as desktops are typically equipped with only speakers but not microphones, and so cannot measure distances on their own. To accommodate such devices, we propose a novel scheme, *DeafBeep*, that relies on estimating the *difference* in the distance between a speaker-only "deaf" device and two other devices equipped with microphones, providing an IDDC for each such triplet. Centaur then uses all the available IDC and IDDC constraints, in combination with WiFi measurements, to estimate the most likely locations of all devices in the office.

Centaur thus combines the ubiquity of RF-based localization with the lower errors of AR techniques to provide superior localization overall. Another key attribute of Centaur is that it is agnostic to the specific RF-localization or acoustic ranging technique used. Thus, it provides the flexibility to pick and combine any RF-based localization scheme with any AR scheme. In summary, our contributions are:

- Centaur: a systematic framework based on Bayesian inference that allows unification of two distinct approaches to localization, RF and AR based schemes.
- EchoBeep: an improved acoustic ranging scheme for non-line-of-sight (NLoS) settings involving a pair of devices, each equipped with both speakers and microphones.
- DeafBeep: a novel scheme for locating a device that is equipped with a speaker but no microphone, based on estimating distance differences.

We present evaluation made through microbenchmarks and deployment in a large office building. We show the improved robustness of EchoBeep in NLoS settings and the ability of DeafBeep to estimate distances differences with a speaker-only device. We show that, by combining RF (WiFi) and AR measurements (including distance difference estimates), Centaur not only improves the localization accuracy over that attainable with any one of the schemes, it also enables localization of devices, such as speaker-only PCs, that were hitherto not amenable to either WiFi-based localization or to acoustic ranging.

2. RELATED WORK

We briefly survey related work, focusing on the WiFi and acoustic techniques most closely related to ours.

2.1 Wireless LAN-based Localization

Wireless LAN-based techniques can be broadly classified as fingerprint based and model-based. Fingerprint based techniques employ a training phase to associate an RF fingerprint (typically RSSI-based) with each of a set of known locations, where the fingerprint could either be expressed in a deterministic form (e.g., RADAR [2]) or in a probabilistic form (e.g., as in Horus [16]). To perform location lookup, the measured RSSI values are compared with the fingerprints recorded previously, to find the closest match.

On the other hand, model-based techniques look to avoid the need for extensive measurements during the training phase by employing a model for RF propagation and estimating the model parameters through a limited set of measurements, either made at the clients (e.g., EZ [6]) or at the APs (e.g., WiGEM [7]).

While wireless LAN-based techniques have been used to localize WiFi-enabled devices such as smartphones and laptops, these sometimes suffer from localization errors of several meters and are also inapplicable to devices such as PCs that are not WiFi-enabled.

2.2 Acoustic Ranging and Localization

Acoustic ranging is attractive because the relatively slow speed of sound (about 332 m/s at sea level) means that the time-of-flight of an acoustic signal, and hence the distance between the transmitter and the receiver, can be measured precisely. In some systems (e.g., Active Bat [9], Cricket [12]), localization is based on the "thunder-and-lightening" principle, wherein a much faster RF signal is used to first synchronize the transmitter and the receiver, and thereafter the one-way time-of-flight of the much slower acoustic signal is measured, to estimate distance. These systems have typically used special-purpose ultrasound hardware on the clients as well as in ceiling-mounted units to enable precise distance measurement with a strong, line-of-sight acoustic signal. In contrast, Centaur focuses on existing devices such as laptops and PCs, typically in locations (e.g., on desks) where the line-of-sight acoustic signal could be blocked by obstructions.

Beep-Beep [11] introduced a novel way to do acoustic ranging, without requiring the devices to be synchronized, which we discuss further in Section 3. However, Beep-Beep suffers from a couple of limitations: it is targeted at proximate devices, where the direct acoustic signal is strong, and it assumes devices that include both a speaker and a microphone, both of which assumptions might not hold in an office environment. We address these limitations in Sections 4 and 5.

Acoustic information has also been used for purposes other than ranging, e.g., for acoustic fingerprinting of locations [13].

2.3 Graphical Model for Localization

Madigan et al. [10] introduced the idea of using a Bayesian graphical model for indoor localization based on WiFi measurements. They introduced various models to incorporate prior knowledge of WiFi signal propagation, thereby reducing the dependence on empirical measurement. Centaur draws inspiration from such prior work, but the specific Bayesian graphical models it employs incorporate the geometric constraints arising from acoustic range measurements, in addition to WiFi constraints. These geometric constraints also render the Gibbs sampling technique employed in [10] impractical, as discussed in Section 9.

2.4 Multimodal Localization

There has also been work on combining wireless and acoustic localization. WALRUS [4] provides room-level localization by having an in-room PC (anchor, in our terminology) emit an ultrasound pulse to signal its presence while concurrently transmitting the identity of the room over wireless. While attractive because of its simplicity, the room-level focus could break down, either because of the non-availability of a PC to serve as the beacon (e.g., in a lecture room) or because the "room" is large (e.g., an open cubicle setting). Centaur avoids these difficulties by using ranging (not just proximity detection) and doing so with respect to both anchor and non-anchor devices.

The Berkeley Multimodal Localization project [1] includes a scheme to combine WiFi-based localization with acoustic localization [14]. This scheme depends on the availability of microphones in devices to enable distance measurement and also requires each of the two approaches — WiFi-based and audio-based — to produce its own location estimate. In contrast, Centaur can accommodate devices without microphones and also inter-device distance and distance difference measurements even if these (partial) measurements by themselves do not permit localization.

3. PRIMER ON LOCALIZATION

3.1 WiFi-based Localization

WiFi-based localization relies on the variation of RF signal strength with location. There are two phases in a typical WiFi-based localization scheme. First, there is a *training phase*, wherein data is collected from various known locations to capture the RSS characteristics of the various APs in the target (indoor) environment. Second, there is the *localization phase* in which a device desiring to locate itself records RSS from various WiFi APs and then uses this measurement coupled with the RSS characteristics determined in the training phase to locate itself. We shall now describe two WiFi-based localization schemes that we make use of in this paper.

HORUS [16]: The basic HORUS system employs an RF fingerprinting based approach to localization. During the training phase, HORUS captures the RSS characteristics of the indoor space through a probability distribution, $P(rss_{AP_k} = r|\mathbf{x} = \mathbf{x}_i)$, which is the probability of seeing an RSS value of r from access point AP_k at location $\mathbf{x}_i$. During the localization phase of HORUS, a device obtains a vector of RSS measurements, $\mathbf{R} = < r_1, r_2, \cdots, r_m >$, where r_i is the RSS from AP_i. The joint probability of observing $\mathbf{R}$ at a location $\mathbf{x}_i$ is computed as,

$$P(\mathbf{R}|\mathbf{x} = \mathbf{x}_i) = \prod_k P(rss_{AP_k} = r_k|\mathbf{x} = \mathbf{x}_i) \quad (1)$$

Bayesian inference is then used to compute the posterior, $P(\mathbf{x} = \mathbf{x}_i|\mathbf{R})$. The final location estimate can then be either the location with the highest probability (*maximum likelihood location*) or computed as an expectation over all locations (*expected location*).

EZ [6]: In contrast, EZ uses an RF modeling based approach to localization. It avoids the need for extensive manual measurement using crowdsourcing without requiring active user participation. Most of these measurements are from unknown locations while a few are from known locations, e.g., near windows where a GPS lock can be obtained. EZ processes the RSS measurements from both known and unknown locations to estimate the locations of the observed APs and the parameters of the log-distance path loss (LPDL) model for each such AP. Localization is then performed using multi-lateration, by converting the RSS measurements to distances from the corresponding APs.

3.2 Acoustic Ranging based Localization

The basic idea here is to perform localization using acoustic ranging with respect to multiple landmarks in known locations. A straightforward way of implementing ranging between two devices A and B is as follows: device A transmits a sound pulse at time t_A and say device B receives the sound pulse at t_B. The time difference $\Delta t_{AB} = t_B - t_A$ then represents the time required by the sound pulse to traverse the distance between device A and B. The distance d_{AB} between the devices is then $c\Delta t_{AB}$, c being the speed of sound.

The above scheme has three key implementation challenges. First, *acoustic signal detection i.e.,* device B must accurately identify the precise point where device A's sound pulse reached B even in the presence of ambient noise. Second, *precise time synchronization i.e.,* the clocks of devices A and B must be in precise synchronization; each 3 ms error in time-synchronization between A and B's clocks results in about 1 m error. Third, *software delays*, e.g. delays due to OS multi-tasking could be large and highly variable, so kernel support is typically required to record the exact time of arrival of the sound pulse.

The first challenge is typically addressed by using sound sequences with very sharp auto-correlation properties (*e.g.,*chirp, Pseudo Random (PN) sequences) agreed upon a priori by both devices A and B. Device B, then determines the exact point of reception of device A's transmission by correlating it with the known sound sequence in its received sound samples. While prior work have addressed the issues of time-synchronization and software delays through the use of special hardware/software support, BeepBeep [11] circumvents both these problems altogether by having both devices A and B transmit sound pulses to each other and then using the sampling rate of the sound-card to keep the measure of time. BeepBeep thus allows acoustic ranging to be implemented as an application without requiring any special kernel support or additional hardware.

BeepBeep [11] : Figure 1 depicts the typical operation of BeepBeep. Initially, both devices A and B turn on their microphones and begin to record ambient sounds. First, device A emits a known chirp signal. Then, after an arbitrary wait, device B emits another chirp signal. Both devices correlate the recorded sequence with the known chirp signal and determine the sample number at which the chirp was recorded. In Figure 1, N_X^Y represents the sample number at which device Y's microphone detected the chirp emitted by device X's speaker. Note that since devices A and B could have started recording on their respective devices at arbitrary times, sample counts across the two devices will not be synchronized. The key idea in BeepBeep is to estimate propagation delay by only using differences in sample counts at each device instead of absolute sample counts. Given the sampling frequency F of the sound card, BeepBeep computes the propagation delay Δt_{AB} in seconds as,

$$\Delta t_{AB} = \frac{\Delta N}{F} = \frac{1}{2F}\left[(N_B^A - N_A^A) - (N_B^B - N_A^B)\right] \quad (2)$$

Challenges with BeepBeep and our solution: There are two key, practical challenges that BeepBeep faces. First, we find that the acoustic signal detection used in BeepBeep performs poorly in non-line-of-sight (NLoS) scenarios, which is quite common in office environments. To address this challenge, we devise an enhanced version called *EchoBeep*, which uses an improved signal detection scheme better suited for NLoS scenarios (Section 4). Second, some devices are only equipped with a speaker but not a microphone (e.g., desktop PCs). To accommodate the participation of such devices in acoustic localization, we device a novel scheme, *DeafBeep* (Section 5), which adapts the ideas in BeepBeep to es-

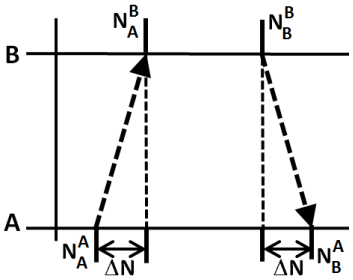


Figure 1: Beep Beep Ranging

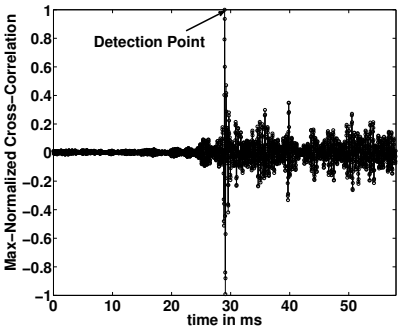


Figure 2: Correlation in BeepBeep for LoS

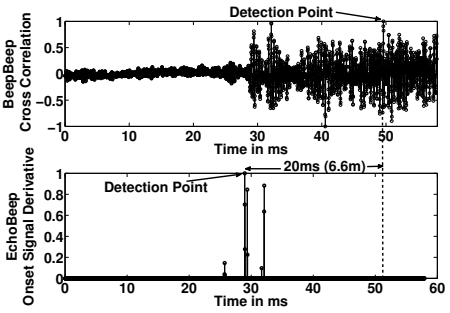


Figure 3: Comparison of EchoBeep and Beep-Beep for NLoS

timate distance differences rather than absolute distances. These distance difference measurements are then used for localization.

4. ECHOBEEP: DISTANCE CONSTRAINTS

While BeepBeep provides extremely accurate ranging, to within a few cm, in LoS environments, we find that it performs poorly in NLoS environments, often resulting in more than 200% errors. In a typical office environment, where devices are often placed on desks within cubicles separated by wooden partitions and obstructed by other clutter, NLoS is a very common scenario. Our analysis reveals that the key reason for the inaccuracy is that the acoustic signal detection component in BeepBeep performs poorly in multipath environments, in the absence of LoS. To overcome this problem, we have developed *EchoBeep*, an enhanced version of BeepBeep that uses an improved acoustic signal detection scheme.

Acoustic signal detection in BeepBeep: In BeepBeep, devices transmit an a priori agreed upon chirp signal (which is a signal with a linearly increasing frequency) to each other. The receiver performs a normalized cross-correlation on the received signal with the known chirp signal in order to detect the arrival of a chirp signal. Figure 2 shows the result of this cross-correlation in a typical LoS scenario, where devices were 6m apart. As seen in Figure 2, the cross-correlation (normalized by maximum) exhibits a very sharp correlation and is easy to detect.

In contrast, Figure 3 depicts the cross-correlation for two devices 6m apart in an NLoS scenario. As seen from Figure 3, unlike the LoS scenario, there are several large peaks in the cross-correlation due to the multiplicity of paths that sound takes to reach the receiver. Furthermore, the sound signal arriving via the shortest path is typically weaker than the reflections because the former has to traverse through obstructions. For example, in Figure 3, the maximum correlation occurs almost 20ms after the arrival of the shortest path signal, which leads to a large error (i.e., overestimation) in the estimation of the time of flight of the acoustic signal. To overcome this problem, BeepBeep looks for shorter peaks that are within a 100-samples window of the tallest peak and at least 85% as tall as the tallest peak. However, there are large errors even with this heuristic.

Acoustic signal detection in EchoBeep : While EchoBeep also uses a chirp signal and cross-correlation at the receiver just as BeepBeep does, it performs additional processing on the normalized cross-correlation to determine weak correlation peaks reliably. The intuition is as follows: the receiver goes from (a) receiving noise to (b) receiving the first, albeit weak, copy of the chirp signal via the direct path, and finally (c) receiving the strongest copy of the chirp signal via a reflected path. When cross-correlation is performed, the difference between the weak signal and noise (i.e., (b) and (a)) would likely be much larger than that between the strongest signal and the weak signal (i.e., (c) and (b)).

Accordingly, let $C(n)$ be the normalized cross-correlation of the received signal with the chirp. EchoBeep performs two additional processing steps on $C(n)$. First, it computes the *onset signal* $O(n)$ given by,

$$O(n) = \max\left(C(k)\right) \forall n \geq k > n - W. \tag{3}$$

$O(n)$ has the property that it captures sudden increases in the cross-correlation (e.g., when going from (a) to (b)) while ignoring the decreases. W is a window that must be larger than twice the duration of the chirp signal (50ms in our implementation). Then, in the second step, we compute the first derivative of $O(n)$ given by,

$$\Delta O(n) = O(n) - O(n-1) \tag{4}$$

This first derivative, makes the transition from noise to the first, albeit weak, peak (i.e., (a) to (b)) stand out compared to subsequent transitions to higher peaks (e.g., (b) to (c)). Thus, this procedure enables the reliable detection of the first peak and hence the shortest acoustic path.

Figure 3 shows an example of $\Delta O(n)$ for the same NLoS setting as in case of BeepBeep. We observe that the first jump in cross-correlation stands out clearly as a spike in $\Delta O(n)$. In order to detect the correct spike, EchoBeep maintains an estimate of maximum height of spike s seen in the absence of a chirp signal (i.e., when cross-correlating with noise) and uses $2s$ as the threshold to eliminate spurious spikes. The earliest non-spurious spike is then deemed as the point of arrival of the chirp. As described in Section 6, the ranging errors in EchoBeep are in the order of 50cm-1m in a typical NLoS office setting compared to an error of 3-5m with BeepBeep. The more accurate ranging provides tighter distance constraints for the Centaur framework.

5. DEAFBEEP: DISTANCE DIFFERENCE CONSTRAINTS

As described in Section 3.2, to estimate the distance between two devices A and B, BeepBeep and EchoBeep require both A and B to be equipped with speaker and microphone each. In a typical office environment, however, devices such as desktop PCs are equipped with only a speaker but no microphone. To allow such speaker-only devices to participate in localization, we have developed a novel scheme called *DeafBeep*. Instead of estimating the distance between two devices, DeafBeep estimates the *difference* in distances of a speaker-only device from two devices that have

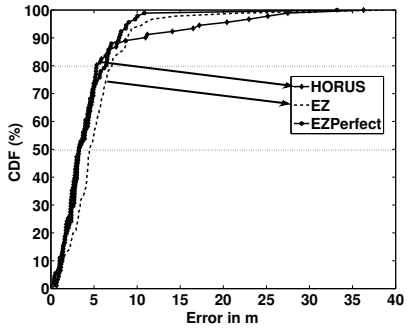


Figure 4: Benchmarking WiFi-based localization

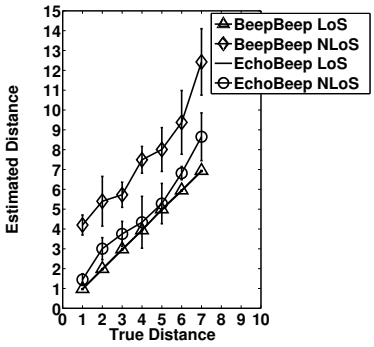


Figure 5: Benchmarking for Acoustic Ranging

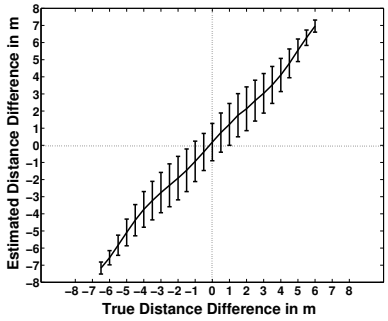


Figure 6: Distribution of Distance Difference Errors

both a speaker and a microphone. Centaur then uses these distance difference constraints to estimate device locations.

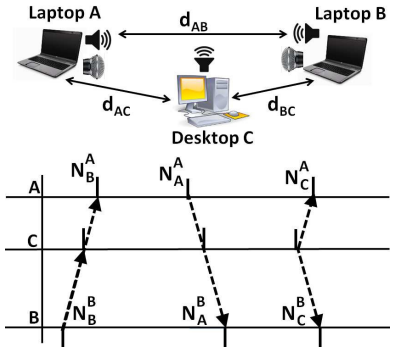


Figure 7: Functioning of DeafBeep

The most basic configuration in DeafBeep comprises three devices located within audio range of each other: one speaker-only device (Desktop C) and two other dual-mode devices equipped with speaker and microphone each (Laptops A and B) as depicted in Figure 7. As in BeepBeep, laptops A and B turn on their microphones and start recording ambient sounds. Then, each of these three devices transmits a chirp sequence at a different point in time. Laptops A and B then analyze the collected sound samples at their respective microphones and detect the reception of each of the three chirp signals using the acoustic signal detection technique used by EchoBeep (Section 4).

In Figure 7, Laptop X detects a chirp from device Y at sample number N_Y^X. Note that, since all three devices could have started recording on their respective microphones at arbitrary times, their sample counts will not be synchronized in time. Let Δt_{XY} be the propagation time for sound to travel from device X to device Y. Given the sampling rate F of the sound cards, DeafBeep then estimates the difference in propagation delays of Desktop C from the Laptops A and B as,

$$\Delta^2_{ABC} = (\Delta t_{AC} - \Delta t_{BC}) = \frac{1}{F}\left[\begin{array}{c}(N_C^A - N_C^B) - \\ \frac{1}{2}\left[(N_B^A - N_B^B) + (N_A^A - N_A^B)\right]\end{array}\right] \quad (5)$$

This equation follows from applying a BeepBeep-like computation to two pairs — AC and BC — and then factoring out the unknowns (i.e., the reception times at C, which are not known because C does not have a microphone). We present a derivation in Appendix A.

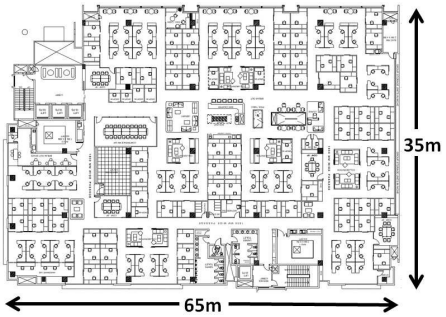


Figure 8: The large office floor used for evaluations

Each distance difference relationship Δ^2_{ABC} results in a geometric constraint between the locations of devices A, B and C, which Centaur then uses to estimate and refine the location estimate of various devices as described in Section 8.

6. MICROBENCHMARKS

In this section, we evaluate WiFi-based localization, acoustic ranging, and acoustic distance differencing individually, to set the stage for Centaur (Section 8), where we apply these techniques in unison. Our evaluation is performed on a large office floor measuring 65m × 35m (Figure 8).

6.1 WiFi Localization

Our training data set was generated by collecting RSS measurements at 117 grid locations spanning the entire floor, with an approximate spacing of 3m between locations and about 12000 beacons gathered per AP at each location. For testing, we measured WiFi RSS at a separate set of 91 test locations within the floor, with just a few tens of beacons gathered per AP.

We evaluated three different WiFi-based localization schemes: HORUS, EZ, and EZPerfect. For HORUS, we used the training data set to create the $P(rss_{AP_i} = r|\mathbf{x})$, i.e., the probability of seeing a certain RSS value r dBm at a location $\mathbf{x}$ within the floor. For EZ, out of the 117 training locations in all, 24 locations near the windows were chosen as known locations since we could obtain a GPS lock there, and the rest were deemed as unknown. Finally, to determine a best-case bound on the accuracy of EZ, we treat all 117 training locations as known and called this EZPerfect.

Figure 4 depicts the Cumulative Distribution Function (CDF) of the localization error of these three schemes over the 91 test

points. HORUS and EZPerfect had a median error of 3.3m while EZ had a median error of about 4.5m. The 80%ile errors for HORUS, EZPerfect, and EZ are 5.3m, 6.4m and 6.9m, respectively. Around the 90%ile mark, the error shoots up to 15-40m for all three schemes. As we discuss later, the incorporation of acoustic ranging constraints in Centaur helps eliminate this long tail in the distribution of localization error.

6.2 Acoustic Ranging using EchoBeep

We tested BeepBeep and EchoBeep in both LoS and NLoS scenarios when devices were placed at 1m to 10m from each other. For LoS scenarios for each distance, we tested BeepBeep and EchoBeep at 90 different locations within the office space by keeping two laptops in LoS of each other. As seen from Figure 5, *both BeepBeep and EchoBeep range extremely accurately with error under 10cm even at a distance of 10m for LoS scenarios*.

To evaluate BeepBeep and EchoBeep in NLoS scenarios, we placed pairs of laptops on tables inside cubicles, separated by various distances through wooden partitions. For each distance we repeated the experiment at 90 different location pairs within the office. As seen from Figure 5, BeepBeep's accuracy degenerates significantly in NLoS environments and can result in an error of over 200% at short distances of 1-3m and errors of up to 100% at greater distances. On the other hand, EchoBeep performs significantly better, yielding an error of 0.5-1m over distances of 1-8m. Note that, in Figure 5, although the *mean* distances estimated by BeepBeep appear to be merely shifted with respect to those estimated by EchoBeep, the former estimates have a much higher *variance*, reflecting the diversity of NLoS paths measured.

6.3 Acoustic Distance Difference

Figure 6 depicts the distribution of errors in the distance difference estimated using DeafBeep over 900 different location triplets corresponding to 3 participating devices in NLoS conditions. The variance of errors is maximum at about $\pm$1.5m when the true distance differences are close to zero, but it drops to about $\pm$50cm as magnitudes of true distance difference increases to 8m. This trend stems from the fact that a distance difference $\Delta^2_{ABC} = 0$, *i.e.,* $d_{AC} - d_{BC} = 0$ can occur in many more ways (e.g., the permissible values are $d_{AC} = d_{BC} \in (0, 10)$) than $\Delta^2_{ABC} = 8$ (e.g., the permissible values are $d_{AC} \in (8, 10)$ and $d_{BC} \in (0, 2)$, or vice versa). Note that the above ranges are bounded by 10m, which is the reliable detection range with 50ms chirps.

7. CENTAUR – EXAMPLE SCENARIOS

Centaur accommodates both *anchors*, whose locations are known a priori (e.g., desktop PCs assigned to specific employees), and *non-anchors*, whose locations need to be estimated (e.g., laptops, temporarily allocated PCs). Centaur uses probabilistic reasoning to combine WiFi measurements with geometric constraints (i.e., inter-device distances and inter-device distance differences) obtained from applying acoustic techniques with respect to known anchors as well as amongst non-anchors, to estimate or refine the non-anchor devices' locations. We present some simple examples to provide an intuition for how Centaur works.

7.1 Simple Example

Consider two laptops A and B that are located within audible distance of each other at unknown locations $\mathbf{x}_A$ and $\mathbf{x}_B$. For ease of exposition in this example, we shall consider locations to be 1D rather than 2D or 3D. Each of these laptops is equipped with a WiFi card, a speaker, and a microphone. Consequently, these can each perform WiFi localization independently and also perform acoustic ranging to estimate the distance d_{AB}, between them.

Using its WiFi RSS measurements ($WiFi_A$), laptop A computes $P(\mathbf{x}_A = \mathbf{x}|WiFi_A)$ – the probability that location of laptop A is $\mathbf{x}$ based on the RSS measurements $WiFi_A$. Similarly, laptop B computes $P(\mathbf{x}_B = \mathbf{x}|WiFi_B)$. These distributions are depicted in Figure 9. As depicted in Figure 9, based on $WiFi_A$ alone, laptop A can lie anywhere between 0m to 5m; its location estimate (maximum likelihood/expected) being around 2.5m. Similarly, based on $WiFi_B$ alone, laptop B can lie anywhere between 9m to 14m; its location estimate being around 11.5m. By performing acoustic ranging, the laptops learn that the distance between them $d_{AB} = 5m$. This distance constraint further limits the possible locations of A and B to $\mathbf{x}_A \geq 4m$ and $\mathbf{x}_B \leq 10m$. Figure 9 also depicts $P(\mathbf{x}_A = \mathbf{x}|WiFi_A, WiFi_B, d_{AB})$ (the probability distribution of laptop A's location based on $WiFi_A$ and the acoustic ranging measurement d_{AB}) and $P(\mathbf{x}_B = \mathbf{x}|\ WiFi_B, WiFi_A, d_{AB})$. The new location estimates of the laptops A and B are thus, 4.5m and 9.5m respectively. *Thus, acoustic ranging information can help decrease the uncertainty inherent to WiFi localization, thereby increasing localization accuracy.* While in this example we assumed that d_{AB} was measured exactly, in general d_{AB} would also have an estimation error, as described in Section 6. In practice, therefore, Centaur also models d_{AB} as a random variable while performing probabilistic inference.

This basic idea can be easily extended to several laptops on a large floor as depicted in Figure 10. Pairs of laptops that are within audible range of each other perform acoustic ranging to obtain a set of distance constraints shown as arrows between pairs of laptops. These distance constraints are then used by Centaur to refine the location estimates of laptop locations obtained from WiFi measurements.

7.2 Composite Example

In this example we try to show the various scenarios that arise in Centaur in a single example.

As depicted in Figure 11 there are two laptops (A and B), and three desktop PCs (C, D and E). The laptops are each equipped with WiFi, speaker and a microphone. Desktops C and D belong to permanent employees and their locations are know a priori and thus are to be treated as anchors, while Desktop E's location is not known a priori. Desktop C has both a speaker and a microphone, whereas desktops D and E have only a speaker on them. All devices are within audible range of each other.

As in the simple example above, the laptops can estimate their locations using WiFi measurements along with the inter-device distance d_{AB}. In addition, since desktop C is an anchor and has both a speaker and a microphone, laptops A and B can further measure distances d_{AC} and d_{BC}. In other words, laptop A must lie on a circular band (with the width of the band dictated by the ranging error) of radius d_{AC} around desktop C. Thus, knowing the location of C, A can further narrow down its location and similarly so can B. Desktop D only has a speaker and so Centaur uses DeafBeep to estimate the distance differences Δ^2_{ABD}, Δ^2_{ACD} and Δ^2_{BCD}. Note that each of these distance difference constraints is associated with a probability distribution of error. Since the locations of C and D are known, Centaur uses these constraints to further narrow down the locations of A and B. Finally, the location of Desktop E is unknown. Since E is only equipped with a speaker, it can estimate the distance difference constraints Δ^2_{ABE}, Δ^2_{ACE} and Δ^2_{BCE}. Using the probability distributions of the locations of A and B and the exact location of C, E uses these geometric constraints to locate itself.

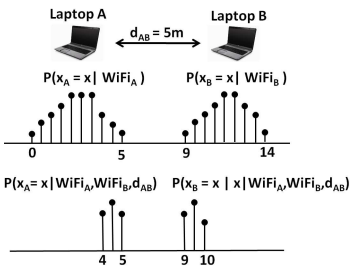


Figure 9: Simple example: probabilistic inferencing for two-laptop case

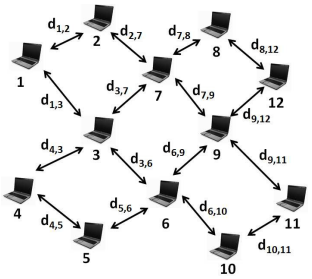


Figure 10: Example with several laptops on a floor

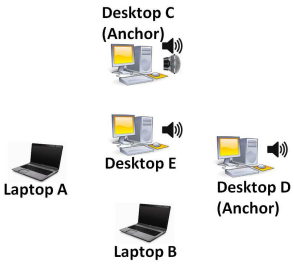
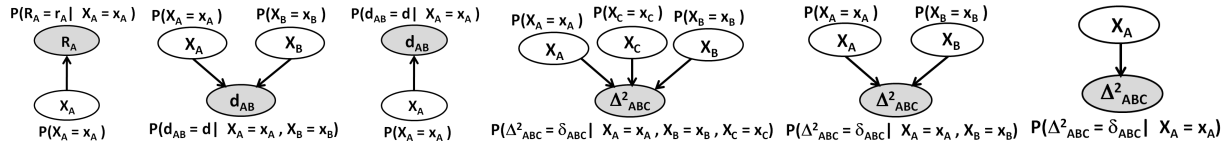


Figure 11: Composite example with anchors, dual-mode, and speaker-only devices



(a) *WiFi* (b) *Range measurements between non-anchors* (c) *Range measurements to an anchor* (d) *DD measurement with non-anchors* (e) *DD measurement with a single anchor* (f) *DD measurement from two anchors*

Figure 12: Graphical models for various measurements in Centaur

In summary, as conveyed in these examples, Centaur uses all manner of WiFi measurements, inter-device distance estimates and inter-device distance difference estimates in a single probabilistic inference framework, to estimate the location of devices.

8. BAYESIAN NETWORKS IN CENTAUR

Centaur uses probabilistic inference to locate devices using the available WiFi measurements, and the inter-device distance and distance difference estimates obtained from acoustic ranging. To accomplish this, Centaur first constructs a Bayesian graphical model to capture all the WiFi measurements and the geometric constraints arising from acoustic measurements. After constructing the Bayesian graph, Centaur infers the most likely set of locations for the devices that satisfies both the geometric constraints and the WiFi measurements. Here we describe Centaur's construction of the Bayesian graph, and then discuss its inferencing technique in Section 9.

8.1 Bayesian Graphs

In a Bayesian graph, random variables are modeled as nodes. In Centaur, all unknown device locations, WiFi measurements, and distance as well as distance difference measurements represent nodes in the graph. Random variables that can be measured are called *evidence* variables (or nodes). So WiFi, distance, and distance difference measurements are the evidence variables in Centaur. Each non-evidence node is also associated with a *prior* — the probability distribution of the variable in the absence of any evidence (i.e., measurements). Finally, directed edges between nodes in the graph represent the relationships between them in the form of conditional probabilities. In the rest of this section, we describe how each measurement is modeled in Centaur as a Bayesian sub-graph and present an example that combines all the sub-graphs together to form the complete Bayesian graph.

8.2 WiFi RSS Measurements

The graph for a WiFi measurement by device A is depicted in Figure 12(a). The prior for location of device A, $P(\mathbf{X}_A = \mathbf{x}_A)$, is the probability, in the absence of any evidence, that the location $\mathbf{X}_A$ is equal to $\mathbf{x}_A$. Centaur uses as prior, a uniform distribution over the entire indoor space. In other words, in the absence of any measurements, a device is equally likely to be located anywhere in the floor. The edge connecting the WiFi measurement node to the location of device A is characterized by the probability distribution, $P(\mathbf{R_A} = \mathbf{r}_A \mid \mathbf{X_A} = \mathbf{x}_A)$, which is the probability of a certain RSS vector $\mathbf{r}_A = < rss_{AP_1}, \cdots, rss_{AP_n} >$ (rss_{AP_k} being the RSS from AP_k) being measured by device A when it is located at $\mathbf{x}_A$. As described in Section 3.1, this distribution is constructed using training data.

8.3 Distance Between Non-Anchors

The graph in Figure 12(b) depicts an EchoBeep measurement d_{AB} between devices A and B. If the locations $\mathbf{X}_A$ and $\mathbf{X}_B$ of devices A and B are $\mathbf{x}_A$ and $\mathbf{x}_B$ respectively, then distance d^{true}_{AB} would be the Euclidean distance between them. EchoBeep however is not free from errors (Figure 5), so it will measure a distance d characterized by the probability distribution $P(d_{AB} = d|\mathbf{X}_A = \mathbf{x}_A, \mathbf{X}_B = \mathbf{x}_B)$.

8.4 Range to an Anchor

If device B is an anchor then $\mathbf{x}_B$ is a known constant. Consequently, the graphical model for this measurement is the same as that for range between non-anchors except with the node corresponding to B removed since it is not a random variable. Figure 12(c) depicts this graph characterized by the probability distribution $P(d_{AB} = d|\mathbf{X}_A)$.

8.5 Distance Difference between Non-anchors

If devices A, B and C are located at $\mathbf{x}_A$,$\mathbf{x}_B$ and $\mathbf{x}_C$, then the distance difference $\Delta^{2true}_{ABC} = d^{true}_{AB} - d^{true}_{AC}$. DeafBeep can be used to estimate this distance difference. The graphical model for this constraint is depicted in Figure 12(d) and is characterized by $P(\Delta^2_{ABC} = \delta_{ABC}|\mathbf{X}_A = \mathbf{x}_A, \mathbf{X}_B = \mathbf{x}_B, \mathbf{X}_C = \mathbf{x}_C)$.

8.6 Distance Difference with Anchors

In a distance difference measurement Δ^2_{ABC}, when one or more devices are anchors, their locations are treated as constants. The graphical model in such cases, is exactly the same as the graphical model for the distance difference constraint for non-anchor devices, but with the anchor devices removed. Figures 12(f) and 12(e) are examples corresponding to distance difference measurements with one and two anchors, respectively.

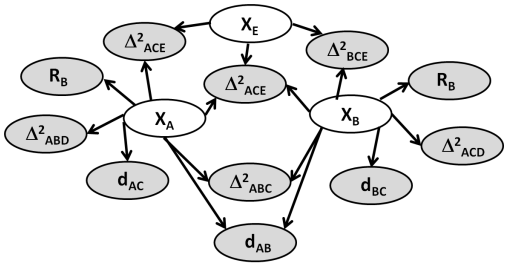


Figure 13: The Bayesian graph for composite example in Section 7

8.7 Putting it all together

Having constructed all the subgraphs for the measurements, Centaur simply combines these together to obtain the overall graph. As an example, Figure 13 depicts the Bayesian graphical model for the composite example described in Section 7.

9. INFERENCE IN CENTAUR

Once the Bayesian graph has been constructed as described in Section 8, Centaur infers the most likely locations for all the non-anchor devices given all the measurements. In this section we describe how Centaur performs this inference on the Bayesian graph.

9.1 Inference in Centaur is NP-Hard

Let $\mathbf{G} = (\mathbf{X}, \mathbf{\Phi}, \mathbf{E})$ be a Bayesian graph, where $\mathbf{X} = \{\mathbf{X}_i\}_{i=1}^{i=N}$ are all the non-evidence variables (nodes), $\mathbf{\Phi} = \{\mathbf{\Phi}_i\}_{i=1}^{i=M}$ is the set of all evidence nodes and $\mathbf{E}$ is the set of all edges in the graph. *The inference problem in G is to determine the posterior probability* $P(\mathbf{X}_i = \mathbf{x}_i|\mathbf{\Phi})$, $\forall i = 1, \cdots, N$, – the probability that a non-evidence variable $\mathbf{X}_i$ takes a value $\mathbf{x}_i$ given all the measured values for non-evidence variables. This probability can be expressed as,

$$P(\mathbf{X}_k = \mathbf{x}_k|\mathbf{\Phi}) \propto \sum_{\forall \mathbf{x}_i, i \neq k} P\left(\begin{array}{l} \mathbf{X_1} = \mathbf{x}_1, \cdots, \mathbf{X_k} = \mathbf{x}_k, \\ \cdots, \mathbf{X_N} = \mathbf{x}_N \end{array} |\mathbf{\Phi} \right). \tag{6}$$

Here, $P(\mathbf{X}|\mathbf{\Phi}) = P\left(\mathbf{X_1} = \mathbf{x}_1, \cdots, \mathbf{X_N} = \mathbf{x}_N\right)$ is the joint distribution of the all non-evidence variables given all the measurements. This joint distribution can then be expressed as,

$$P(\mathbf{X}|\mathbf{\Phi}) = \prod_{\forall \mathbf{X}_i \in Ch(\mathbf{G})} P(\mathbf{X}_i = \mathbf{x}_i|\mathbf{\Psi}(\mathbf{X_i})) \prod_{\forall \mathbf{X}_i \in \mathbf{X}} P(\mathbf{X}_i = \mathbf{x}_i) \tag{7}$$

In Eqn 7, $Ch(\mathbf{G})$ is the set of all variables that have at least one parent (i.e., are child nodes in the Bayesian graph), and $\Psi(\mathbf{X}_i)$ is the set of all of node $\mathbf{X}_i$'s parents. Further, all evidence variables in Eqn 7 are set to their measured values.

Inference in Bayesian graphical networks is a well studied area. When a Bayesian graph can be represented as a Poly-tree (i.e., the undirected version of the graph has no loops), the Pearl's message passing algorithm [3] solves the inference problem with $O(N)$ complexity, N being the number of nodes in the graph. For a general graph with loops, however, inference has been proven to be NP-Hard [3]. The graphs generated in Centaur almost always have several loops as seen in Figure 13, consequently exact inference in Centaur is an NP-Hard problem.

There are three popular approaches to approximate inference in Bayesian networks – loopy belief propagation [3], sampling based techniques [3] and variational methods [3]. We found that using loopy belief propagation on graphs constructed by Centaur often neither converged nor yielded correct results. We further discovered that the large number of geometric constraints in a typical Bayesian graph in Centaur makes it particularly ill-suited to using sampling techniques such as Gibbs sampling [3] and importance sampling, and required impractically large running time for convergence. This is because, the large number of geometric constraints limits the feasible set of device locations to a very narrow region in the $2N$ dimensional space of possible device locations that is hard to draw samples from. Finally, we found that the distribution of RSS measurements at several locations is non-Gaussian and not amenable to an analytical form tractable to variational methods. Consequently, we devised an approximate inference technique specifically tailored to the graphs generated by Centaur.

9.2 The Maximum log-likelihood approach

An alternative to solving the inference problem is be content with finding device locations that maximize the log-likelihood of the joint distribution, $\log P(\mathbf{X}|\mathbf{\Phi})$. This is especially suitable in Centaur since Centaur only attempts to estimate the device locations. The maximization problem is also NP-Hard and requires searching over all possible combinations of values of $\mathbf{X}$. Further, the log-likelihood function in Centaur is a multi-modal function and is not amenable to iterative schemes such as Newton-Raphson or gradient decent that rely on finding a local maxima. One approach is to use a Genetic Algorithm (GA) to search efficiently. However, we found that the GA took a long time to converge to a solution. This is for the same reason that sampling-based inference techniques perform poorly in the context of Centaur, as discussed previously.

9.3 The Centaur Approach

In Centaur, we found that a hybrid, two-step approach of search and inference performs the best. In the first step, *partial inference step*, Centaur performs an exact inference using Pearl's algorithm, accommodating as much evidence as possible without having loops in the graph. The goal of this step is to narrow down the space of possibilities and make maximum log-likelihood search tractable. The second step, *maximum log-likelihood step*, then takes a maximum log-likelihood approach by searching over space of possibilities given by the first step. In the rest of this section, we describe this scheme.

9.3.1 Partial Inference Step

The key idea in this step is to partition the entire Bayesian graph $\mathbf{G}$ into a sequence of loop-free subgraphs $< \mathbf{G}_1, \mathbf{G}_2, \cdots, \mathbf{G}_n >$. The key property in the partition is that no two sub-graphs have any common evidence nodes and the union of all evidence nodes across all sub-graphs covers the entire set of evidence nodes in $\mathbf{G}$. Let $\mathbf{\Phi}_i$ be set of all evidence nodes in sub-graph $\mathbf{G}_i$. The scheme starts by performing exact inference on $\mathbf{G}_1$. Then for each successive graph $\mathbf{G_k}$, the results of the inference performed on the preceding sub-graphs, $\mathbf{G_1}, \cdots, \mathbf{G_{k-1}}$, is used as the prior to perform exact inference on $\mathbf{G}_k$. The scheme is algorithmically described below.

1: $PartialInference()$
2: $< \mathbf{G}_1, \cdots, \mathbf{G}_k >= Partition(\mathbf{G})$.
3: Set all prior $P(\mathbf{X}_i)$ in $\mathbf{G}_1$ to the same as $\mathbf{G}$.

4: Compute $P_{\mathbf{G}_1}(\mathbf{X}_i|\mathbf{\Phi}_{\mathbf{G}_1})$ $\forall i$ using Pearl's algorithm.
5: **for** $k = 2$ to n **do**
6: Set all priors $P(\mathbf{X}_k)$ in $\mathbf{G}_k$ to the same as $P_{G_{k-1}}(\mathbf{X}_i|\mathbf{\Phi}_{\mathbf{G_{k-1}}})$.
7: Compute $P_{\mathbf{G}_k}(\mathbf{X}_i|\mathbf{\Phi}_{\mathbf{G}_k})$ using Pearl's algorithm.
8: **end for**

1: $Partition()$
2: $\mathbf{G}_{rem} = \mathbf{G}$.
3: $k = 1$
4: **repeat**
5: Find the set of edges $\mathbf{E}_k$ that need to be removed to eliminate cycles from $\mathbf{G}_{rem}$.
6: Remove all evidence nodes associated with these edges in $\mathbf{E}_k$ to obtain $\mathbf{G}_k$.
7: Remove all evidence nodes considered in $\mathbf{G}_k$ from $\mathbf{G}_{rem}$.
8: **until** $\mathbf{G}_{rem}$ has not evidence nodes

The complexity of $Partition()$ is $O((|E| + N)N)$. The complexity of the loop in $PartialInference()$ is $O(|E|)$. This is because, at each step in the loop, the complexity of Pearl's algorithm is $O(|E_i|)$, $|E_i|$ being the number of edges in G_i. Since sum of all edges across all the subgraphs is equal to $|E|$, i.e., the number of edges in the graph G, the complexity of the loop is $O(|E|)$. In a practical setting, since the running time of $Partition()$ is insignificant compared to that of the Pearl's algorithm, the practical complexity of $PartialInference()$ is $O(|E|)$.

9.3.2 *Maximum log-likelihood Step*

In this step we run a GA to determine the set of device locations that maximize the log-likelihood. In all our settings we found that the GA typically converges in under a minute (as evaluated in Section 10). This is because the partial inference step reduces the space of possibilities dramatically.

10. RESULTS

Given its goal of combining acoustic and WiFi localization in synergy to improve localization error, we evaluate Centaur with a view to answering the following key questions under various deployment scenarios:

- How much can WiFi localization benefit from acoustic distance and distance difference measurements?
- How much can acoustic distance difference and distance measurement based localization in turn benefit from WiFi measurement?
- How well can we use Centaur to locate speaker-only devices with no microphone or WiFi?
- What are the typical running times for Centaur?

We implemented and deployed Centaur in a large office floor depicted in Figure 8 (Section 6). Based on our deployment, we try and quantitatively answer the above questions.

10.1 Implementation and Deployment

The Centaur software comprises a Centaur server and a client. The clients were deployed on devices such as laptops and desktop PCs. The clients connect to the server whenever the devices are turned on and report to the server as to whether they have a WiFi-card, speakers and/or a microphone. In case of anchor nodes, the anchors additionally report their known locations to the server. The Centaur server then, depending on the capabilities of the device, schedules the devices to perform WiFi measurements, transmit Chirp sequences, and record ambient sounds for EchoBeep and DeafBeep measurements. The clients transmit all recorded data (WiFi measurements and recorded sound samples) to the server. Centaur then uses EchoBeep and DeafBeep to convert the recorded samples to inter-device distance and distance difference estimates. It then constructs the Bayesian graph (Section 8) and uses the inferencing algorithm described in Section 9.3 to estimate the most likely locations of all devices.

10.2 Exp I: WiFi localization with acoustic ranging among non-anchors

In this experiment we ask the following question, *"In the absence of any anchors (devices with known locations), can the performance of WiFi localization be improved by using inter-device distance measurements?"* This scenario corresponds to the simple example (Figures 9 and 10) in Section 7. First, laptops were placed at various locations within the floor and their locations were estimated only using WiFi localization (HORUS). Then, pairs of laptops estimated their inter-device distances using EchoBeep. A laptop can, however, only estimate distance to another laptop if it is within audible range. So in successive experiments, we increased the number of laptops that any given laptop could estimate distance with using acoustic ranging. When there are N laptops in audible proximity of each other, there are $\frac{N(N-1)}{2}$ distance measurements among the N laptops to be combined with N WiFi measurements made at the individual laptops. Figure 14 depicts the CDF of the localization error, with N increasing from 1 to 5 laptops and compares it to the case where laptops used on WiFi localization.

As seen from Figure 14, even with just one other laptop in audible range, there is an improvement in localization performance compared to using only WiFi measurements. The 50%ile error decreases from 4.2m to 3.4m while the 80%ile error decreases from 7m to 5m. The most significant improvement occurs in the tail of the localization errors. Errors as large as 20m that occurred in WiFi are completely eliminated with the help of just one acoustic distance measurement. This is because, the likelihood that both devices will have a large localization error at the same time is low and the node with a high WiFi localization error benefits from the other. As we increase the number of laptops within audible proximity, the performance further improves. The 50%ile error reduces to 2.4m and the 80%ile error reduces to 4m. However, increasing the number of proximate laptops beyond 3 does not bring significant improvements. *This experiment demonstrates how WiFi localization error can benefit from inter-device distance estimation even when there are no known locations.*

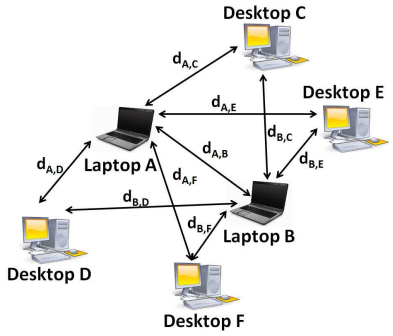


Figure 17: Setup for evaluating fusion of WiFi with speaker-only anchor devices

10.3 Exp II: WiFi localization with acoustic ranging to speaker-only anchor devices

In a typical office environment the location of several desktop PCs can be known a priori and thus these devices can be treated as

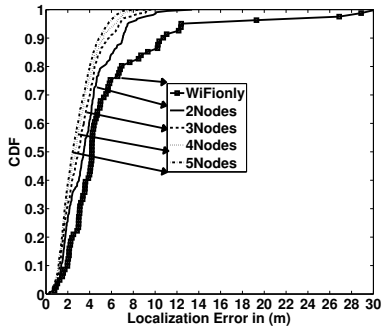


Figure 14: Performance of WiFi localization with ranging among non-anchors

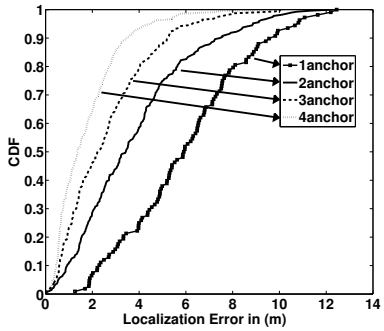


Figure 15: Performance of localization with speaker-only anchors

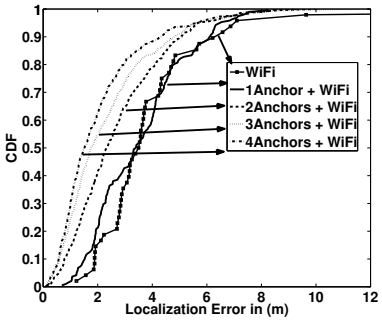


Figure 16: Performance of localization with WiFi and speaker-only anchors

anchors. However, most desktop PCs are typically equipped only with a speaker but no microphone. These desktop PCs however can aid in localization by measuring acoustic distance difference measurements using DeafBeep. In this section we first ask the question, *"How much can WiFi localization benefit from distance difference measurements?"*

In order to answer this question, we use the setup depicted in Figure 17. Since at least two laptops are required to measure distance difference, our set up comprises of two laptops and N ($N = 1,\cdots,5$) desktop devices equipped with only speakers. We evaluated this set up in 41 different configurations spread across the entire office floor.

First, we quantify the performance of distance difference localization by itself, by locating the laptops using distance difference measurements only. The distribution of localization errors for different number of anchors is depicted in Figure 15. As the number of anchors increases from 1 to 5, the 50%ile and 80%ile errors localization error decreases from 6m to 1m and 8m to 2m respectively. Next, we use Centaur to combine WiFi measurements at the laptops with the distance difference measurements. As seen from Figure 16, using distance difference information from 2 or more anchors, WiFi localization benefits significantly. While the 50%ile and 80%ile errors for WiFi itself are 3.6m and 4.4m respectively, they decrease to 1m and 2.4m with the help of 4 speaker-only anchors. Another interesting observation comparing Figures 15 and 16 is the fact that even acoustic localization benefits from WiFi measurements. The errors of using 1,2 and 3, anchors without aid of WiFi measurements are higher than those that result from using WiFi measurements. For example, the 50%ile error for localization using 3 anchors by themselves is 2.4m and decreases to 1.8m when combined with WiFi. *These experiments indicate that both WiFi localization and acoustic localization benefit in terms of increased localization accuracy by combining each other's information.*

10.4 Exp III: WiFi localization with acoustic ranging to anchors with speaker and mic

In the previous set of experiments anchors were equipped with only speakers and could be used only in measuring distance differences. In these experiments we ask the question, *"How can WiFi and acoustic localization gain from each other if acoustic ranging to anchors was possible?"* In order to answer this question we used the configuration depicted in Figure 18 and tested it at several locations across the office floor.

First, we evaluate the performance of only acoustic localization without WiFi. Figure 19 depicts the distribution of localization errors with increasing number of anchors. As seen from Figure 19,

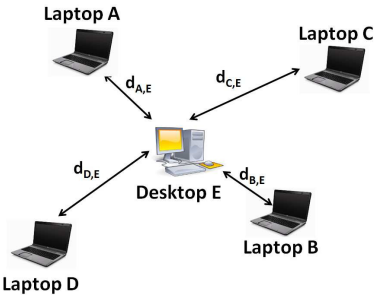


Figure 18: Setup for evaluating fusion of WiFi with ranging to anchor devices

ranging based localization significantly outperforms distance difference based localization or WiFi based localization, with 50%ile and 80%ile errors of 60cm and 80cm respectively. This is expected since acoustic ranging is extremely accurate. Next, we evaluate the performance of Centaur when WiFi and acoustic ranging information are combined. Figure 20 depicts the distribution of localization errors with increasing number of anchors. As seen from Figure 20, there is a significant improvement in WiFi performance with the aid of acoustic ranging even to a single anchor and improves further with increasing number of anchors. However, the performance of the combination of WiFi and acoustic ranging is almost the same as using only acoustic ranging. In fact we found that there was only a 1% decrease in average error by adding WiFi measurements to acoustic range measurements. *Hence, we conclude that given acoustic range measurements to anchors, adding WiFi information does not bring any significant improvement in localization performance.*

10.5 Exp IV: Locating speaker-only devices without any anchors

In some cases it may be required to locate desktop PCs which are equipped with only a speaker but no microphone. If there are laptops equipped with both WiFi cards they can locate themselves and then use distance difference localization to locate the desktop PC. Note that these speaker only devices cannot be localized in any other way without combining WiFi and acoustic measurements. In these experiments we ask the question, *"How accurately can we locate speaker-only devices using devices that are equipped with WiFi-cards, speaker and a microphone?"* The experimental setup for these experiments is the same as depicted in Figure 18 except that, the desktop PC is equipped with a speaker but no microphone and its location is unknown. Thus, the laptops measure distance dif-

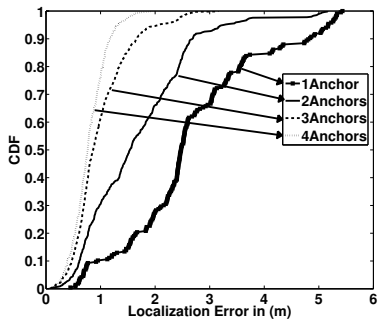


Figure 19: Performance of acoustic localization by ranging to anchors.

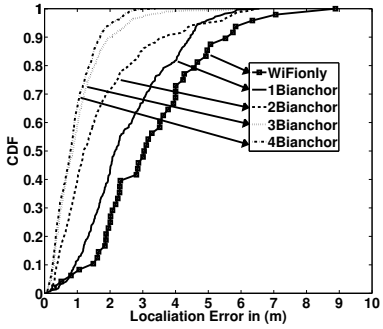


Figure 20: Performance of WiFi + acoustic localization by ranging to anchors

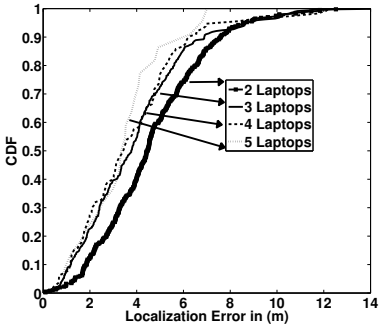


Figure 21: Performance of localizing speaker-only devices without any anchors

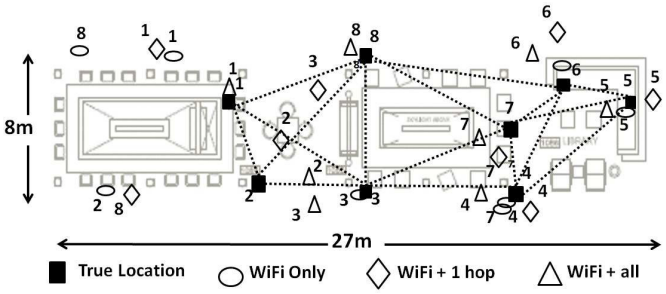


Figure 22: Locating 8 dual-mode devices (laptops), using WiFi only, WiFi coupled with AR with respect to 1-hop neighbors, and WiFi coupled with all available AR information.

ferences to the desktop, while locating themselves using WiFi and acoustic ranging to each other. Figure 21 depicts the distribution of localization errors of the desktop PC with increasing number of laptops in audible proximity. As seen from Figure 21, the 50%ile and 80%ile errors are 2m and 4m respective when the speaker only device has 4 laptops in its audible proximity.

Device	Error in m		
	WiFi Only	WiFi + 1 Hop	WiFi + All
1	4.5	4.0	0.7
2	6.3	2.1	2.9
3	0.7	4.9	3.5
4	0.7	1.1	2.4
5	0.7	1.2	1.6
6	1.3	3.5	2.4
7	4.1	1.5	2.8
8	13.2	10.7	1.4

Table 1: Location estimation errors seen for Experiment V

10.6 Exp V: Locating several dual-mode devices simultaneously

"How will Centaur perform in a realistic setup, where devices are typically distributed in various cubicles of an office floor, not all within acoustic range of each other?" In order to answer this question we conducted an experiment in a realistic setting comprising 8 laptops located across the floor. Figure 22 depicts the true locations of all the laptops (indicated as solid squares). Laptops that are within acoustic range of each other are connected by dotted lines in Figure 22 and are able to range to each other.

In order to evaluate Centaur's performance in this setting, we consider three different scenarios: (i) WiFi-only, where each node is localized only using its own WiFi measurements, (ii) WiFi + 1-hop, where WiFi localization is combined with AR with respect to 1-hop acoustic neighbors, and (iii) WiFi + all, where WiFi localization is combined with all available AR information across the 8 nodes.

Figure 22 depicts the final locations of the devices obtained for each of the three scenarios, while Table 1 provides the exact values of the localization error (in meters). We observe that WiFi-only localization can result in a significant error: 4.5m, 6.3m, and 13.2m, respectively, for nodes 1, 2, and 8. Adding AR constraints, whether from the 1-hop neighborhood or from the entire network, helps reign in these outliers.

In the WiFi+1-hop case, we compute a location estimate for each node separately, by fusing its WiFi location estimate with AR constraints relative to neighboring nodes that it is within acoustic range of. For example, as shown in Figure 22, the 1-hop location estimate for node 8 is computed by fusing the WiFi-based location estimate of 8 with the WiFi-based location estimates of, and the acoustic range relative to, each of its acoustic neighbours: 1, 2, 3, and 7. As the figure shows, these additional constraints, even from the local neighborhood alone, cause the location estimate of node 8 to be pulled in close to the node's true location, with its localization error decreasing from 13.2m in the WiFi-only case to 10.7m in the WiFi + 1-hop case. However, the grossly incorrect location estimate for node 8 coupled with the AR constraint between it and node 3 also causes the latter to be pulled away from its true location (and towards the incorrect location estimate for 8), with the localization error for node 3 increasing from 0.7m in the WiFi-only case to 4.9m in the WiFi + 1-hop case.

When we consider all the WiFi and AR constraints and solve for the locations of all nodes simultaneously (the WiFi + all case), we generally obtain the best results. The large error due to outliers is eliminated (e.g., the localization error for node 8 drops to 1.4m) and no error is worse than about 3.5m (node 3 at 3.5m has the worst error across all nodes).

Thus, Centaur's approach of holistically fusing WiFi and AR information across a set of nodes is beneficial, even when not all nodes are within "earshot" (i.e., acoustic range) of each other.

10.7 Running Time of Centaur

Finally, we try and answer the question, *"How long does Centaur require to perform localization?"*. Table 2 shows the running

times for Centaur for various configurations. As described in Section 9, the running time of centaur is $o(e)$ complexity, e being the number of measurements (acoustic as well as WiFi). Table 2 provides the times taken by a Intel Xeon CPU E5405 at 2.0 GHz with 2 processors, 8 GB RAM and a 64 bit operating system in running each experiments I through IV. As seen from Table 2, typical running times for experiments I through III are within a few seconds. However, the running time of experiment IV is a few minutes. Analysis indicated that generating the probability distributions for distance difference constraints in the absence of anchors was the key computation intensive operation in experiment IV. Each distance difference constraint without anchors generates a family of hyperbolae across the floor. Having no analytical means of computing these distributions required us to generate them using sampling techniques that lead to the larger computation time. Experiment V, which corresponds to a realistic scenario, took 15.9s. These are reasonable times to locate devices in an office environment for asset tracking.

Configuration	No Edges	No of Nodes			
		2	3	4	5
Exp I	$N + \frac{N(N-1)}{2}$	2.8s	5.0s	9.0s	15.1s
Exp II	$3 + 2N$	1.2s	1.9s	2.6s	3.7s
Exp III	$1 + N$	2.8s	2.9s	3.4s	5.4s
Exp IV	$2N + \frac{N(N-1)}{2}$	45.8s	80.7s	179s	285s

Table 2: Running times for various experiments

11. CONCLUSION

We have presented Centaur, which provides a novel, Bayesian framework for combining WiFi-based localization with the geometric constraints arising from acoustic ranging. We have also presented two new acoustic techniques — EchoBeep for ranging in non-line-of-sight settings, and DeafBeep for locating speaker-only devices. Through extensive experiments in an office environment, we have shown the effectiveness of EchoBeep and DeafBeep. Further, Centaur was able to eliminate instances of large WiFi based localization error arising from RSS variation.

12. ACKNOWLEDGEMENTS

We thank our shepherd, Moustafa Youssef, and the anonymous reviewers for their constructive comments.

13. REFERENCES

[1] The Berkeley Multimodal Loalization Project. http://mmle.icsi.berkeley.edu/mmle/project.html.
[2] P. Bahl and V. N. Padmanabhan. RADAR: An Inbuilding RF-based User Location and Tracking System. In *INFOCOM*, 2000.
[3] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2007.
[4] G. Borriello, A. Liu, T. Offer, C. Palistrant, and R. Sharp. WALRUS: Wireless Acoustic Location with Room-Level Resolution using Ultrasound. In *MobiSys*, 2005.
[5] Y.-C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm. Accuracy Characterization for Metropolitan-scaleWi-Fi Localization. In *MobiSys*, 2005.
[6] K. Chintalapudi, A. P. Iyer, and V. N. Padmanabhan. Indoor Localization Without the Pain. In *Mobicom*, 2010.
[7] A. Goswami, L. E. Ortiz, and S. R. Das. WiGEM : A Learning-Based Approach for Indoor Localization. In *CoNEXT*, 2011.
[8] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Practical Robust Localization over Large-Scale 802.11 Wireless Networks. In *MobiCom*, 2004.
[9] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The Anatomy of a Context-Aware Application. In *Mobicom*, 1999.
[10] D. Madigan, E. E., R. P. Martin, W. Ju, P. Krishnan, and A. Krishnakumar. Bayesian Indoor Positioning Systems. In *Infocom*, 2005.
[11] C. Peng, G. Shen, Z. Han, Y. Zhang, Y. Li, and K. Tan. A beepbeep ranging system on mobile phones. In *SenSys*, 2007.
[12] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *MobiCom*, 2000.
[13] S. Tarzia, P. Dinda, R. Dick, and G. Memik. Indoor Localization without Infrastructure using the Acoustic Background Spectrum. In *MobiSys*, 2011.
[14] O. Vinyals, E. Martin, and G. Friedland. Multimodal Indoor Localization: An Audio-Wireless-based Approach. In *IEEE Semantic Ccmputing*, 2010.
[15] A. Ward, A. Jones, and A. Hopper. A New Location Technique for the Active Office. *IEEE Per. Comm.*, 4(5):42–47, 1997.
[16] M. Youssef and A. Agrawala. The Horus WLAN Location Determination System. In *MobiSys*, 2005.

APPENDIX

A. DERIVATION OF DEAFBEEP

Suppose that devices A and B are equipped with a microphone and a speaker while device C is equipped with only a speaker but no microphone. As depicted in Figure 23, the microphones of devices A and B recording samples at times t_0^A and t_0^B respectively. Each of the three devices now takes turns to transmit an acoustic chirp signal at times t_1^A, t_1^B and t_1^C respectively. As depicted in Figure 23 device X receives this signal from device Y at time t_Y^X (the time of arrival is determined using EchoBeep described in Section 4). Let the distance between between devices X and Y be d_{XY} (d_{XX} is the distance between the speaker and microphone of the same device X). Further, let the speed of sound be V. Then,

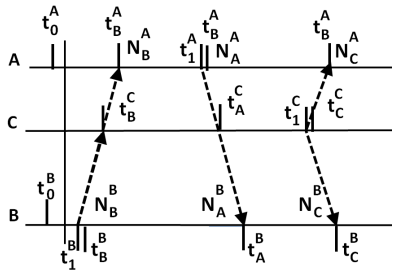


Figure 23: Derivation of DeafBeep

$$N_Y^X = S\left(t_1^X - t_0^X + \frac{d_{XY}}{V}\right) \tag{8}$$

In Equation 8, $S = \frac{1}{F}$ is the sampling rate of the sound card in samples/sec and F its sampling frequency in Hertz. From Equation 8, we obtain,

$$N_Y^A - N_Y^B = S\left[\frac{d_{AY} - d_{BY}}{V} + \left(t_1^A - t_0^A\right) - \left(t_1^B - t_0^B\right)\right] \tag{9}$$

From Equation 9 we can deduce that,

$$\begin{aligned}&\left[N_C^A - N_C^B\right] - \tfrac{1}{2}\left[\left(N_A^A - N_A^B\right) + \left(N_A^A - N_A^B\right)\right] \\ &= \tfrac{S}{V}\left[\left(d_{AC} - d_{BC}\right) + \left(d_{AA} - d_{BB}\right)\right]\end{aligned} \tag{10}$$

Typically, d_{AA} and d_{BB} will each be very small (in the order of a few centimeters). Further, if device A and B are identical, $d_{AA} = d_{BB}$. Consequently, for all practical scenarios, $d_{AA} - d_{BB}$ will be very small if not almost zero. Equation 5 in Section 5 thus, directly follows from Equation 10.