LoSeCo: Location-based Search Computing for Pervasive Device Augmentation

Yiqiang Chen†, Zhuo Sun†, Juan Qi†, Derek Hao Hu‡, Qiang Yang‡
†Pervasive Computing Center, Institute of Computing Technology, Chinese Academy of Sciences
Email: {yqchen, sunzhuo, qijuan}@ict.ac.cn
‡Department of Computer Science and Engineering, Hong Kong University of Science and Technology
Email: {derekhh, qyang}@cse.ust.hk

***Abstract*—Understanding human intention and performing different activities automatically is one of the key problems in pervasive computing. In this paper, a new location-based search computing framework (LoSeCo) is proposed to allow one's pervasive device to augment search devices. The objective of our problem is to recognize the real-time user goal through goal inference from traditional Wi-Fi localization techniques. We use accelerometer-based tracking to reduce the effort we need to collect Wi-Fi signals and save battery power consumption effectively. With the help of short-range search, the goal recognition module is enhanced, compared to previous "location-only" approaches. Therefore, we could augment our mobile devices by automatically analyzing our needs and connecting to corresponding devices. Experimental results on real-world wireless network environments validate the effectiveness of our approach and that even a rough localization accuracy can meet the need of QoS (Quality of Service) in search computing behaviors.**

I. Introduction

It is expected that in future pervasive computing environments, we could use Internet services anytime and anywhere with only a mobile phone at hand. Such a problem is rather challenging despite the fact that one may think it is just a matter of squeezing the PC's functionalities into a mobile phone. Essentially, it needs another round of thought of a new computing infrastructure from Internet servers, ranging from desktops, handheld devices and computer networks. Optimally, one would hope such a new infrastructure should at least contain owned devices (OD), generic private computing devices (PD), shared computing devices (SD) and the "cloud" with much computing and storage resources (CS). And the infrastructure described should be able to use other kinds of devices for augmentation, for more efficiency and effectiveness.

In this paper, we would focus on the problem of OD augmentation in this infrastructure, based on our LoSeCo technology. OD augmentation means that your device can be adaptive to changes by changing its functions according to the perceived demands of human users. The challenges in OD augmentation research not only lie in how to make functional augmentations in hardware or software parts, but also in that the OD should know how to adapt itself. For example, a question is: When should our mobile phone be tuned to connect to a TV channel and show the programs? When should our mobile phone be connected to a printer and use the printer services? When should our mobile phone be connected to cloud-computing service centers and search for drivers or other devices? It is not practical to keep it on since it would search for nearby devices continuously, with a large consumption of battery power. The user behavior on mobile phones is likely to be very different from that on desktop computers, where we type many keywords and submit them to search engines. Then, the real-time user's *OD augmentation goal* (what kind of device the user wants its mobile phone to be connected) will be an important issue to explore. We have the following three questions:

1) When does the OD decide to search for nearby devices for function augmentation?
2) How do we infer OD augmentation goal based on both real-time location and nearby device's information?
3) Can the same Quality of Service (QoS) be promised if we can only get rough localization accuracies based on current location estimation technology?

Our proposed LoSeCo framework aims to answer these three questions. First, an accelerometer-based tracking service can be used for the potential OD augmentation by motion detection. Then a Wi-Fi based location estimation module searches for devices nearby through user goal inference. Finally, the OD can be prepared and augmented for specific functions. We implement the LoSeCo system in our wireless environment on a Nokia N95 mobile phone which is equipped with accelerometer, Wi-Fi and Bluetooth. The search computing model is processed by Bluetooth, based on both the motion status detected by the accelerometer and location information inferred from Wi-Fi signals. The experimental results show that LoSeCo is able to provide a better QoS of OD augmentation even under a rough localization accuracy.

We make four main contributions in this paper. Firstly, we utilize the sensors and functionalities integrated in a mobile device without adding extra sensors or devices for detecting user activities, from which we could infer contextual information. Then, an accelerometer-based tracking service is designed based on Wi-Fi, using power-efficient computation, to save a great deal of battery power consumption. Next, unlike the traditional location-aware services that are solely based on location and historical information, LoSeCo considers the possible nearby device information to make our goal recognition module more accurate. Finally, it is demonstrated that our proposed framework can still augment OD effectively under low-accuracy localization cases.

II. Background

A. Location-aware Services

Location-aware service, also called location-based service is an important research issue in the field of mobile computing for many years. It can be defined as services that use the location of the target for adding value to the service. The location information can be considered as a specific kind of context information, which is helpful to adapt the behaviors of service. For example, recommendation of discounted commodities is received when going into a shopping mall, printers are automatically prepared when entering a printing room. As stated before, location information is just adding value to the service to make it suitable and intelligent. Meanwhile, as the location information is often obtained through the handheld devices taken by human, location-aware service is a typical device augmentation application. Our proposed LoSeCo is also a kind of location-aware service for pervasive device augmentation by providing them the ability of automatic search computing.

Location information can be obtained from different ways. Considering the possible interfaces and sensors in our OD, the localization approach can be implemented through GPS module, wireless signals like Wi-Fi and Bluetooth, or acceleration measured by accelerometer. These approaches are available in different environments (outdoor or indoor) and have different localization accuracies.

In outdoor environment, GPS usually performs well by providing localization accuracy within 5 meters. It is unnecessary to get information from other ways since this localization accuracy is enough to track out the trajectory of the object. While GPS is disabled in indoor environment, a feasible and reliable approach is to use the Wi-Fi signals from existing infrastructures to estimate location. Due to the multi-path fading and shadow effect caused by complex indoor environment, traditional line-of-sight triangulation methods for localization can not meet the desired accuracy. Therefore, a lot of localization algorithms based on machine learning have been proposed in this area.

B. Context-aware Services

Context-aware services are also related functions of LoSeCo. Context is any information that can be used to characterize the circumstances of an entity, which can be a person, a place, or an object, therefore, a context-aware system uses the context to provide the relevant information and/or services for the user. One of the goals of context-aware computing is to have the applications do the right thing at the right time for users without their direct manipulation. It is a typical form function augmentation for devices.

C. Related Work

We first introduce some related works on Wi-Fi based localization in this section. This localization approach employs radio signal information obtained from wireless beacons to infer possible location. Early work in this area includes the RADAR system [3] that used a manually calibrated table of signal strength and their nearest neighbor algorithm gave a median spatial error of 2.94 meters. This error was reduced to 2.37 meters using a Viterbi-like algorithm in follow-on work [2]. The Horus system [11] identified different reasons for the wireless channel variations and addressed them to achieve high accuracy. It used location-clustering techniques to reduce the computational requirements of the algorithm and had an error of less than 0.6 meters on average. LEMT algorithm [10] reconstructed a radio map using real-time signal strength readings received at the reference points. This technique effectively accommodated the variations of signal strength over different time periods without the need to rebuild the radio maps repeatedly. Using Place Lab [6], commodity laptops, PDAs and cell phones estimated their positions by listening for the cell IDs of fixed radio beacons, such as wireless access points, and referencing the beacons' positions in a cached database. It achieved 20-30 meters median accuracy in large scale area. The UCSD ActiveCampus [5] project used 802.11 to compute the location of wireless Pocket PCs both indoors and outdoors. Instead of manual calibration, they used a formula that approximated the distance to an AP as a function of signal strength. Using a hill climbing algorithm, their system estimated location to an accuracy of about 10 meters using signal strengths from multiple APs.

We also discuss some related works on context service systems. [7] proposed a new definition and classification of context for mobile and collaborative learning. In [4], the authors model the context dynamics and deploy it for determining the appropriate agent who is best suitable for a task in an enterprise. In [8], by the technology of Touch-And-Play (TAP), users can disclose their context by touching the specific device. So the users with multimedia devices may simply touch them to establish network connection,transfer data, and provide the required service.

III. Proposed Framework - LoSeco

In this section, we will describe our proposed framework LoSeCo into detail. We illustrate the architecture in Figure 1. It is composed of four main components: acceleration detection, location estimation, search computing and device augmentation. Each component collects useful information from its own sensor and passes it down to other components. With enough information accumulated, available services are searched out automatically as feedback to ODs. For example, A person living in the campus always takes his mobile phone and does daily works. The mobile phone is equipped with accelerometer, Bluetooth, Wi-Fi card and so on. The accelerometer is used to discover what the person is doing, and the Wi-Fi card will help to compute where the person is, and the Bluetooth will scan the devices around, and recommend a connection to the necessary devices to realize functional augmentations to ODs. More specifically, these components are described in detail in the following sections.

A. Acceleration Detection

We use accelerometer-based tracking to help us better understand what points are “significant” or change points in a user trace. Such a motivation comes from the limited battery capacity of mobile phones. It is a common approach to collect WiFi signals uninterruptedly at some predefined frequencies, e.g. 1 or 2Hz. However, it is a waste of energy to scan access points periodically when the user is actually not moving. For example, the user may be sitting in the dorm with his computer all the afternoon, we don’t need to repeatedly locate him in such a circumstance. Therefore, the only important information is the “significant” timestamps as well as places. When will the user move? Where is he moving to? Intuitively, even in the route of an activity, most locations are not essential to infer this activity except several significant locations. For instance, to go to dining room for lunch, one may need to leave his office, cross a concourse, take a lift then reach the dining room. If we know he is waiting the lift going down the dining room, we may infer that he is going for lunch with a higher probability without any other information. We define these significant locations as *Critical Points* that have critical meanings for localization and prediction. That is to say, what we need is sparse Wi-Fi signals from critical points.

It is easy for us to understand which points are critical but how does the mobile phone know? To solve this problem, we propose the accelerometer-based tracking approach based on the observation that the motion state of a user usually changes at critical points. For instance, one walks to a lift and then stands to wait for it, or he stops to open the door of his office in order to get in. There is always a motion change at critical points like lifts and doors. We define the location where a motion change occurs as a *Change Point* and show it in Figure 2. Nearly all critical points are change points but it does not hold on the other hand. While it is not easy to find critical points, change points can be easily detected by the accelerometer. The holder’s motion states like stationary, walking, running can be inferred from the acceleration values detected by the accelerometer. Therefore we propose to use change points to replace critical points first and filter those useless change points later. We restrict that only when the accelerometer detects a change point, the device will execute a Wi-Fi scan, hence it can greatly reduce the Wi-Fi scan frequency in a day to avoid battery running out.

To explain how the change point is detected, we show a slice of acceleration values in one axis direction in Figure 2. It is clearly observed that there are different acceleration patterns for different motion states. When staying stationary, the acceleration almost keeps a balanced value with slight changes. While walking, acceleration values fluctuate around the balanced value and positive values appear more in the walking direction. The pattern of running is a little similar with walking, but displays a more intense fluctuation.

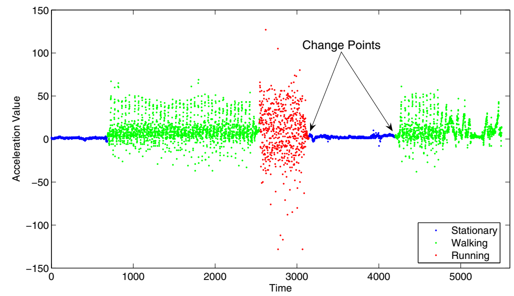

Fig. 2. Acceleration patterns of different motion states.

As the patterns of different motion states vary significantly, we are able to distinguish stationary state and non-stationary state. In each time window, if the current state differs from the previous one, that is a change point. The Wi-Fi signals collected at these change points are fewer than traditional uninterrupted collection methods. Thereby it reduces the power consumption a lot.

B. Location Estimation

Once a change point is detected, the Wi-Fi card scans all the access points around. Assume there are n access points, we get a received signal strength (RSS) vector $V(v_1, v_2, ..., v_n)$ to estimate current location l. It is usually necessary to build a localization model or radio map first based on Wi-Fi training data for the areas. Then we input the RSS vector to estimate the location using localization algorithms. We introduce several commonly used localization algorithms which are applied in our framework.

1) Nearest Neighbor (NN): Nearest-Neighbor is the most commonly used method in Wi-Fi localization area. The collected Wi-Fi training data set includes various signal patterns $\{S_i(s_{i1}, s_{i2}, ..., s_{in})\}$ at different locations, and they can be tabulated in a table as shown in Table I. s_{ij} represents the signal strength collected from j^{th} AP in i^{th} pattern.

TABLE I
A SAMPLE OF OUR RADIO MAP

Location	RSS Vectors			
Office 5531	$AP_1:-49$	$AP_2:-74$	$AP_6:-84$	...
	$AP_1:-50$	$AP_2:-72$	$AP_6:-87$	...
	$AP_1:-49$	$AP_2:-71$	$AP_6:-85$	...
		......		
Lift 19	$AP_{12}:-89$	$AP_{13}:-87$	$AP_{14}:-87$	
	$AP_{12}:-88$	$AP_{13}:-84$	$AP_{14}:-87$	
	$AP_{12}:-89$	$AP_{13}:-86$	$AP_{14}:-88$	
		......		
Concourse	$AP_{29}:-59$	$AP_{30}:-59$	$AP_{31}:-55$	...
	$AP_{29}:-63$	$AP_{30}:-60$	$AP_{31}:-45$	...
	$AP_{29}:-55$	$AP_{30}:-62$	$AP_{31}:-51$	...
	$AP_{29}:-58$	$AP_{30}:-58$	$AP_{31}:-52$	...
		......		
...		......		

To estimate a location, a set of measured distance $\{d_i\}$ between the input RSS vector V and each pattern S_i in the table is calculated as Equation 1. Then the location with the

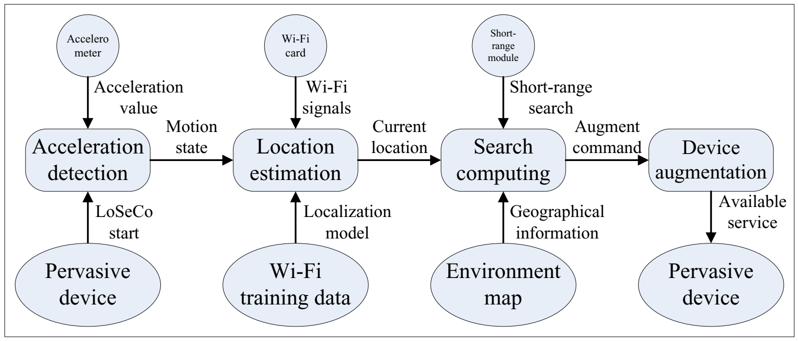

Fig. 1. The architecture of LoSeCo.

smallest distance is assigned as the estimated location. That is, $l = \arg\min_{l_i} d(i)$.

$$d(i) = \sqrt{\sum_{j=1}^{n} (v_j - s_{ij})^2} \tag{1}$$

2) Naive Bayes (NB): Naive Bayes is another commonly used method in this area. The Wi-Fi training data set reflects the signal distributions $P(l_i)$ at different locations l_i. Therefore, for each access point, the signal value distribution $p(V|l_i)$ at each location can be approximated from the dataset. Then NB is applied to calculate the conditional probability $P(l_i|V)$ of the input RSS vector on each location.

$$p(l_i|V) = \frac{p(V|l_i)P(l_i)}{P(V)} \propto p(V|l_i)P(l_i) \tag{2}$$

The prior probability $p(V|l_i)$ and $P(l_i)$ can be calculated from the Wi-Fi training data set. The location with the largest probability is considered as the estimated location. That is $l = \arg\max_{l_i} p(l_i|V)$.

3) Hidden Markov Model (HMM): NN and NB both focus on a snapshot of the signal strength value, whereas if we have a sequence of signal strength values, we could use Hidden Markov Model (HMM) to solve the localization problem. Suppose the data set includes some trace sequences, we can train an HMM which treats user's locations as hidden states and captures the user's underlying characteristics. As shown in Figure 3, an HMM for modeling user traces is defined as a quintuple (L, O, λ, A, π), where L is the location-state space. O is an observation space consisted of sequences of raw Wi-Fi data. λ is the conditional probability of an observation at some state. A is the location-state transition matrix which encodes the probability of moving from one location to another. π is an initial location-state distribution about where a user's initial place. Given some traces as training data, three main parameters (λ, A, π) of an HMM can be learned out by applying the *expectation-maximization* (EM) algorithm. Using the trained localization model online, the most probable location of an observation can be predicted by *Viterbi* algorithm [9].

Regardless of what localization algorithm is used, current location can be estimated by inference from received Wi-Fi signals and it is can be used as input for search computing.

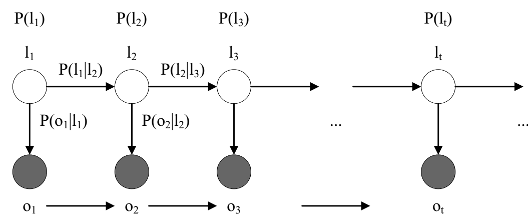

Fig. 3. The Hidden Markov Model of trajectory

C. Search Computing

Search computing is the procedure that searches available services or devices around, and chooses the appropriate one according to the location information. Therefore, the input of estimated location is crucial with two functions. The first is to determine whether it is necessary to search, because not every location is meaningful for search computing. For instance, to print materials, only the final destination, printing room, is the exact place to execute the search command. Although the meeting room we passed by is also significant, it may lead to an incorrect search decision. The second is to determine which particular device we should try to connect.

In fact, the search decision made based on estimated location is a tradeoff between power consumption and search quality. If search decision gives search command at every location, it is equivalent to let it search in many unimportant places and battery power is wasted for most of the time. Therefore, it is important and necessary to extract useful geographical information from the environment map to make correct search decisions. On the other hand, an estimated location may not always be correct and could lead an inappropriate search decision and thereby affect the QoS of search computing by missing some necessary search. To reduce these kinds of errors, the nearby searchable devices' information is gained from some short-range communication modules like Bluetooth, UWB, Zigbee etc and added for decision. For example, when in a meeting room, the accessible devices include a projector, a screen controller and a desktop computer. Only relying on the estimated location, that is meeting room, it is hard to tell which device the user wants to access. With the short-range Bluetooth scan, the device with the best Bluetooth connectivity is the nearest to the user, and it may be the most possible one that the user intends to connect.

We apply association rule mining [1] to discover relations between services and locations. Since the augmentation goal is related to OD's location as well as the connectivity between them, association rules are mined for making decisions. Two important constraints, minimum thresholds on support and confidence, are used to adjust the rule learning and specified by ourselves. Due to space constraints, we omit the detailed discussions here.

D. *Device Augmentation*

When the search decision is made out with a high confidence, device augmentation will give corresponding respond. The most common service is automatic connection to the targeting device, for example, connecting the nearby printer to print materials, connecting the projector to make a presentation and so on. Other services are also available under this framework. When a user gets into his/her office, the email receiver on the OD is shut down and switches to the one on his/her personal computer. Similarly, when he/she is leaving, it will switch back to the OD again. Naturally, there are cases that there will be more than one devices which the OD could connect, under this situation, UI would prompt the users for selection to better understand the user's intention.

IV. Experiments

We implemented our LoSeCo framework in a university environment covered by wireless networks to test the effectiveness of our approach. [1] There are access points deployed at nearly every corner in the university to provide a large-scale coverage, thereby a mobile phone equipped with a wireless card and carried by a user can receive Wi-Fi signals everywhere and we can use such signals to track the user's trajectory. The layout is shown in Figure 4. A student named Jim carries the Nokia N95 mobile phone and helps us collect the WiFi signals when he is moving about in the university campus.

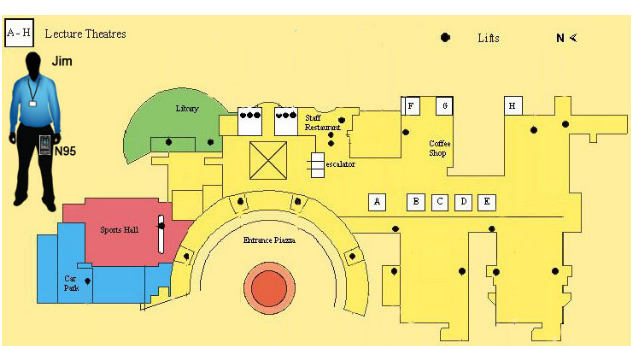

Fig. 4. The layout of our experimental environment.

[1]http://www.ust.hk/images_lib/liftmap.gif

A. *Experimental Setup*

Trajectories of Jim are collected by us during a four-week phase. Examples of traces collected include going to the classroom from lab, going to a lecture hall from a classroom, going to the student canteen for lunch from a lecture hall and so on. A sample of the trace sequence of Wi-Fi signals is shown in Table II. There is a timestamp tagging the time at which such WiFi signals are received, acceleration values detected by the accelerometer, and the collected RSS vectors, which can be used to perform localization. Since we are using a change point-based tracking approach, Wi-Fi signals are collected only at "change points" and therefore most of the remaining entries have empty RSS vectors.

For location inference, a radio map [3] containing the ground truth of Wi-Fi signal distribution is needed. It tabulates typical RSS values collected at the significant places as shown in Table I. It can be seen as a Wi-Fi fingerprint database for estimating locations.

To further validate the performance of device augmentation step in LoSeCo, Jim is asked to take N95 to perform some search computing behaviors, such as go to print materials, go to class and use projectors for presentations and so on. We use Bluetooth connections to automatically connect possible devices from which we could augment N95 and these devices may include printer, desktop computer or projector. Once connection with the device is established, it is considered as a successful search computing behavior and would be counted towards our QoS evaluation in the next section.

B. *Performance Evaluation*

From the description of our LoSeCo framework, the performance evaluation of such a framework needs to be evaluated in two aspects. One is our localization accuracy and the other is QoS of search computing behaviors.

Following conventional standards, localization accuracy is measured at different error distances, i.e. if we allow the distance between the estimated location and the actual location to be less than a predefined threshold value, what is the accuracy of our localization results? Three localization algorithms described in our paper, namely NN, NB and HMM, are compared and the results are shown in Figure 5. We could see that as error distance increases, the localization accuracy increases as well, which seems to fit our intuition. Since we are performing localization in a large-scale environment (school campus), it is reasonable to set error distance to a larger value (e.g. 10 meters), and we could reach a localization accuracy of around 70%. We could also see that HMM performs best amongst the three algorithms we described since it takes sequential information into consideration and NB performs better than NN.

The QoS of search computing is calculated as:

$$QoS = \frac{\text{Number of successful search}}{\text{Number of total search}} \quad (3)$$

We test the QoS performance with different degrees of localization accuracy. The result is shown in Figure 6. It

TABLE II
A SAMPLE OF COLLECTED DATA SET.

Timestamp	Acceleration	RSS Vectors
2008-09-02 112.0	$< 2, 41, -35 >$	These points are not "change points",
2008-09-02 112.0	$< 3, 45, -35 >$	so we have no RSS vectors recorded.
2008-09-02 113.0	$< 4, 59, -33 >$	<Intentionally empty>
2008-09-02 113.0	$< 14, 53, -7 >$	$< AP_1 : -49, AP_2 : -74, AP_6 : -84, AP_7 : -51 >$
2008-09-02 114.0	$< 32, 29, 10 >$	$< AP_1 : -50, AP_2 : -79, AP_3 : -76, AP_6 : -79 >$
2008-09-02 114.0	$< 32, 36, 8 >$	$< AP_1 : -50, AP_2 : -79, AP_3 : -76, AP_6 : -79 >$
2008-09-02 115.0	$< 32, 49, 18 >$	<Intentionally empty>
2008-09-02 115.0	$< 35, 46, 18 >$	<Intentionally empty>

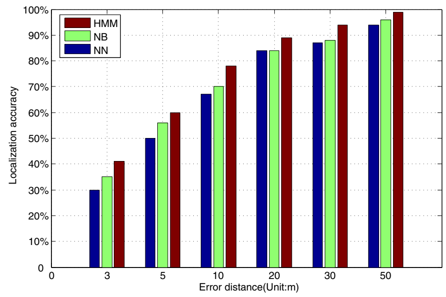

Fig. 5. Localization accuracy within different error distances.

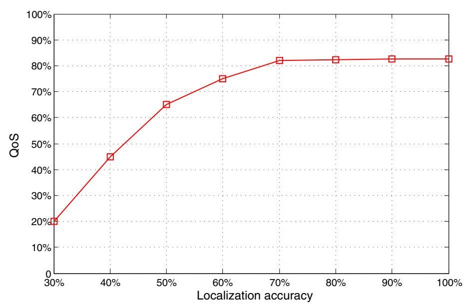

Fig. 6. QoS with different localization accuracy

can be shown that as localization accuracy increases, QoS performance would generally increase as well, and it would converge to around 80%. From Figure 6, we could also see that QoS performance would reach 80% even when the localization accuracy is only around 60%, which suggests that we could guarantee a relatively high QoS performance even when the localization accuracy is low.

It can be analyzed that, when localization accuracy is low, we could not detect our location accurately and would try to connect the wrong devices for augmentation and thus fails to establish connection. However, such a QoS performance might converge since it is difficult for OD to distingush between similar devices in the same location, e.g. if there is a printer and a desktop computer in the same room, which device should N95 try to connect? Such problems are not easy to solve when we only infer the users' location or simply choose the nearest device and we plan to study such problems in our future work.

V. Conclusion and Future Work

In this paper, we propose LoSeCo, a location-based search computing framework that tries to augment pervasive devices to solve the challenging problem of real-time user goal inference in pervasive environments. Our proposed LoSeCo framework aims to utilize both location-aware as well as accelerometer techniques to decide when and how we should search nearby services for augmentation. The information provided by short-range search aids to recognize our goal for augmentation. Without adding any hardware, we demonstrate the effectiveness of our framework on a Nokia N95 mobile phone and that the QoS of search computing can be guaranteed even with only rough localization accuracy. In the future, we plan to extend our current framework towards other devices from which we could augment our OD, e.g. cloud computing and storage resources.

Acknowledgements

The authors would like to thank Qiong Ning and Fangfang Dong for data collecting and processing. This work is supported by National High Technology Development "863" Program of China (Grant No. 2007AA01Z305).

References

[1] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items in large databases," in *SIGMOD Conference*, 1993, pp. 207–216.
[2] P. Bahl and V. N. Padmanabhan, "Enhancements to the radar user location and tracking system," Microsoft Research, Tech. Rep., 2000.
[3] ——, "Radar: An in-building rf-based user location and tracking system," in *INFOCOM*, 2000, pp. 775–784.
[4] M. D. Choudhury, H. Sundaram, A. John, and D. D. Seligmann, "Context aware routing of enterprise user communications," in *PerCom Workshops*, 2007, pp. 39–44.
[5] W. G. Griswold, P. Shanahan, S. W. Brown, R. T. Boyer, M. Ratto, R. B. Shapiro, and T. M. Truong, "Activecampus: Experiments in community-oriented ubiquitous computing," *IEEE Computer*, vol. 37, no. 10, pp. 73–81, 2004.
[6] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. E. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. N. Schilit, "Place lab: Device positioning using radio beacons in the wild," in *Pervasive*, 2005, pp. 116–133.
[7] J. Malek, A. Derycke, and M. Laroussi, "A middleware for adapting context to mobile and collaborative learning," in *PerCom Workshops*, 2006, pp. 221–225.
[8] D. G. Park, J. K. Kim, J. B. Sung, J. H. Hwang, C. H. Hyung, and S. W. Kang, "Context aware service using intra-body communication," in *PerCom*, 2006, pp. 84–91.
[9] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
[10] J. Yin, Q. Yang, and L. M. Ni, "Adaptive temporal radio maps for indoor location estimation," in *PerCom*, 2005, pp. 85–94.
[11] M. Youssef and A. K. Agrawala, "The horus wlan location determination system," in *MobiSys*, 2005, pp. 205–218.