Extracting places from traces of locations

Article *in* ACM SIGMOBILE Mobile Computing and Communications Review · July 2005
DOI: 10.1145/1094549.1094558 · Source: DBLP

Extracting Places from Traces of Locations

Jong Hee Kang[1], William Welbourne[1], Benjamin Stewart[1], Gaetano Borriello[1,2]

[1]Dept. of Computer Science and Engineering, University of Washington, Seattle, WA 98195
[2]Intel Research Seattle, 1100 NE 45th Street, Seattle, WA 98105
{jhkang, evan, stewartb, gaetano}@cs.washington.edu

ABSTRACT
Location-aware systems are proliferating on a variety of platforms from laptops to cell phones. Locations are expressed in two principal ways: coordinates and landmarks. However, users are often more interested in "places" rather than locations. A place is a locale that is important to an individual user and carries important semantic meanings such as being a place where one works, lives, plays, meets socially with others, etc. Our devices can make more intelligent decisions on how to behave when they have this higher level information. For example, a cell phone can switch to a silent mode when the user is in a quiet place (e.g., a movie theater, a lecture hall, or a place where one meets socially with others). It would be tedious to define this in terms of coordinates. In this paper, we describe an algorithm for extracting significant places from a trace of coordinates, and evaluate the algorithm with real data collected using Place Lab [14], a coordinate-based location system that uses a database of locations for WiFi hotspots.

Categories and Subject Descriptors
I.5.3 [**Pattern Recognition**]: Clustering - *algorithms*

General Terms
Algorithms, Experimentation

Keywords
Clustering, Location-aware system, WiFi hotspots

1. INTRODUCTION
Location-aware systems are proliferating on a variety of platforms from laptops to cell phones. In these systems, locations are expressed in two principal ways: coordinates and landmarks. In coordinate-based systems such as GPS, Place Lab [14], and E911, location is specified by coordinates (latitude and longitude in this case). In landmark-based systems a location is represented as a relative proximity to one or more landmark objects. Examples of these systems include those that report the GSM cell towers within range [7] or, on a smaller scale, those that report well-known Bluetooth beacons. Location, expressed in terms of either coordinates or landmarks, is useful for many applications. For example, coordinate-based systems can be used for trip planning and navigation assistance, while landmark-based systems are useful for more local or personal applications, such as finding others that may be in the vicinity of the same landmark [6].

However, users are more interested in "places" rather than locations. A place is a locale that is important to an individual user and carries important semantic meanings such as being a place where one works, lives, plays, meets socially with others, etc. Our devices can make more intelligent decisions on how to behave when they have this higher level information. For example, a cell phone can switch to a silent mode when the user is in a quiet place (e.g., a movie theater, a lecture hall, or a place for personal reflection). A location-based reminder [3] can remind the user of what she has to carry or what she mistakenly left behind based on the user's starting point and likely destination. In a location-based to-do list application [9], the user can associate a to-do list with each place, and the application displays applicable to-do list items as the user moves about and reaches different places where they have an errand to complete. A navigation assistant application for the cognitively-impaired can guide and assist the users in reaching their destination [10].

To translate locations measured by the underlying location sensing technologies into places, we need to define the places of interest in terms of locations. For example, a user's work place can be defined as a rectangular region around her office represented in coordinates. And, if the user's current position reported by her location system is within the region (possibly with some tolerance), she is considered to be at her work place. A simple approach to define places is to define each place by hand. However, manual definition of places does not scale well. Instead, we need an approach that can automatically determine important places. These can be defined as the places where the user spends a significant amount of time and/or visits frequently. There are several parameters to consider in making this determination: duration of a visit to a place, the frequency of visits, the minimum distance between significant places, and the interaction between these three parameters.

In this paper, we describe an algorithm for extracting significant places from a trace of coordinates. In the trace, significant places are the regions where many location measurement samples are clustered together. The algorithm identifies these clusters from the trace automatically. We also evaluate the algorithm experimentally with real traces collected from Place Lab [14], a location system that uses WiFi access points' beacon messages to determine a user's location.

WMASH'04, October 1, 2004, Philadelphia, Pennsylvania, USA.

2. RELATED WORK

To the best of our knowledge, all previous work on place extraction with a coordinate-based location system has been done using GPS. The advantage of GPS is that it is a standardized, globally available location system that can be easily adapted for use in a variety of contexts. Potential drawbacks of GPS include its inability to function indoors, its occasional lack of accuracy due to the geometry of visible satellites, and loss of signal in urban canyons and other "shadowed" areas.

Early work on place extraction with GPS used loss of signal to infer important indoor locations. Marmasse and Schmandt [9] identify a place as a region bounded by a certain fixed radius around a point, within which GPS disappears and then reappears (as in when a user enters and leaves a building). This approach is sufficient to identify indoor places that are smaller than a certain size (e.g., a home), but does not account for larger indoor places (an office complex or convention center), and suffers from false positives (caused by the many possible outdoor GPS shadows).

A similar but more improved approach to extracting important locations is proposed by Ashbrook and Starner [1]. In that work, sets of important coordinates are identified as those at which the GPS signal reappears after an absence of 10 minutes or longer. What is more, these sets are then clustered into significant locations (i.e. places) using a variant of the k-means clustering algorithm. Through this further separation of the notions of coordinate and place, and by using a minimum time bound of 10 minutes, Ashbrook and Starner are able to overcome the place-size limitations and most of the false positives that Marmasse and Schmandt's approach suffers from. However, the use of GPS signal loss to infer place still leaves us unable to infer important outdoor places, or multiple places within a single building.

The statistical inferencing machinery used by Patterson et al. [11] and Liao et al. [8] to learn and predict daily transportation routines from GPS traces is also able to identify important outdoor places within a user's routes. Patterson et. al. use real-world knowledge of bus schedules and stop locations, along with acceleration and turning speed to infer mobile places (e.g. bus, car), as well as the location of parking lots and bus stops where users change mode of transportation. Liao et al. use mode-changes such as GPS signal loss and acceleration peaks to identify frequented locations in a totally unsupervised manner. Though identification of indoor places is still not possible, these approaches offer steps toward a more robust and complete place extraction scheme.

Recently, we have found parallel work by Hariharan and Toyama [5] that uses an approach that is very similar to ours in that they use time information to determine important places. From location histories, they first extract instances of a user spending some time in one place, which they call stays. Then, they cluster the stays and find places where one or more users have experienced a stay, which they call destinations. By using the time information in extracting stays from the location histories, they can better identify semantically important places. However, their algorithm is computationally expensive because it requires distance computations between all pairs of locations within a specified period of time to determine stays. Interestingly, they use similar tuning parameters to our algorithm and set them to almost identical values. This gives us increased confidence in our similar algorithms.

There has also been some recent work in place extraction using a landmark-based location system. Laasonen, et al. [7] use the cells of a GSM phone network to learn important places in a user's daily routine. Their approach does not require any knowledge of network topology or even the locations of the cell towers. This approach allows place extraction over a wide area using existing infrastructure (the cellular network). However, the resolution of the derived places is very coarse (the same as that of a GSM cell – that can reach as far as a few kilometers in range although many are of much smaller range).

3. EXTRACTING PLACES

3.1 Trace of Locations

We use Place Lab [14] to collect traces of locations. Place Lab provides a way for a WiFi enabled client device to automatically determine its location. Place Lab exploits the fact that each WiFi access point periodically broadcasts its unique MAC address as part of its management beacon. Each client device holds a database that maps these addresses to longitude and latitude coordinates. When the client device receives beacon messages from nearby access points, it retrieves each access point's coordinate from the database and computes its location by averaging the locations of the access points (a simple centroid tracking scheme). The accuracy of Place Lab depends on the density and the arrangement of access points. Place Lab gives better location estimation in the areas with a high density of access points. Today, many cities and towns around the world have a high enough density of access points to provide location estimates on the order of 50-100m. Place Lab works best in urban areas – exactly the opposite of GPS which works best in open areas. More importantly, Place Lab works indoors as well where AP density is likely to be high in modern office and even residential environments.

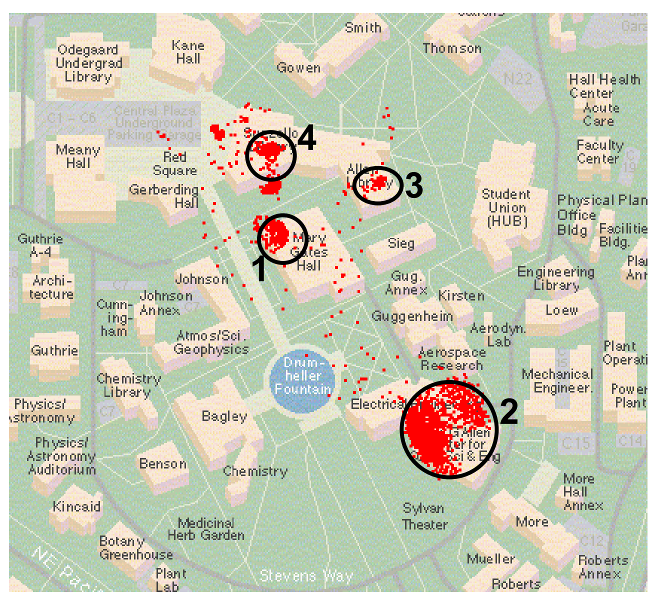


Figure 1. A trace of locations collected with Place Lab.

As with most location systems, including GPS, multiple measurements in the same location do not necessarily yield the same coordinates due to errors and variations. In Place Lab the set of access points that the client device sees in a location can vary, and consequently, the location estimate obtained by averaging the access points' locations varies as well. Thus, the important places where the user spends considerable time appear as clusters of locations in the traces. Figure 1 shows a trace of locations collected by one of the authors. The location was recorded once per second, and each location was represented as a dot in the figure. The author visited four places during the logging period, and those four places are shown as densely clustered regions in the figure.

We need to design an algorithm that will extract these significant places from the traces automatically.

3.2 Existing Clustering Algorithms

Identifying densely clustered regions from the trace is basically a clustering problem, and we first tried two popular clustering algorithms: k-means [4] and Gaussian mixture model (GMM) approach [2]. Figure 2 shows the significant places identified by these clustering algorithms.

Ideally, the system should be able to identify the evolving set of significant places by itself without input from the user. And, at the same time, the system should accurately report if the user is at one of the significant places. However, the existing clustering algorithms are not quite right for these purposes. One of the problems is that they require the number of clusters as a parameter. So, before running the clustering algorithms, the user has to specify the number of important places in advance. Although there are variations of the clustering algorithms that compute the number of clusters automatically [13], they still have other limitations. One of these is that the clusters generated by the clustering algorithms include unimportant locations. As seen in Figure 2, the clusters become unnecessarily larger when they include the intermediate and transitory locations between truly significant places. With this clustering result, the system could report that the users are in one of the significant places when they are actually merely in transit between them. Another limitation is that these clustering algorithms require a significant amount of computation and may not work well for small battery-powered mobile devices.

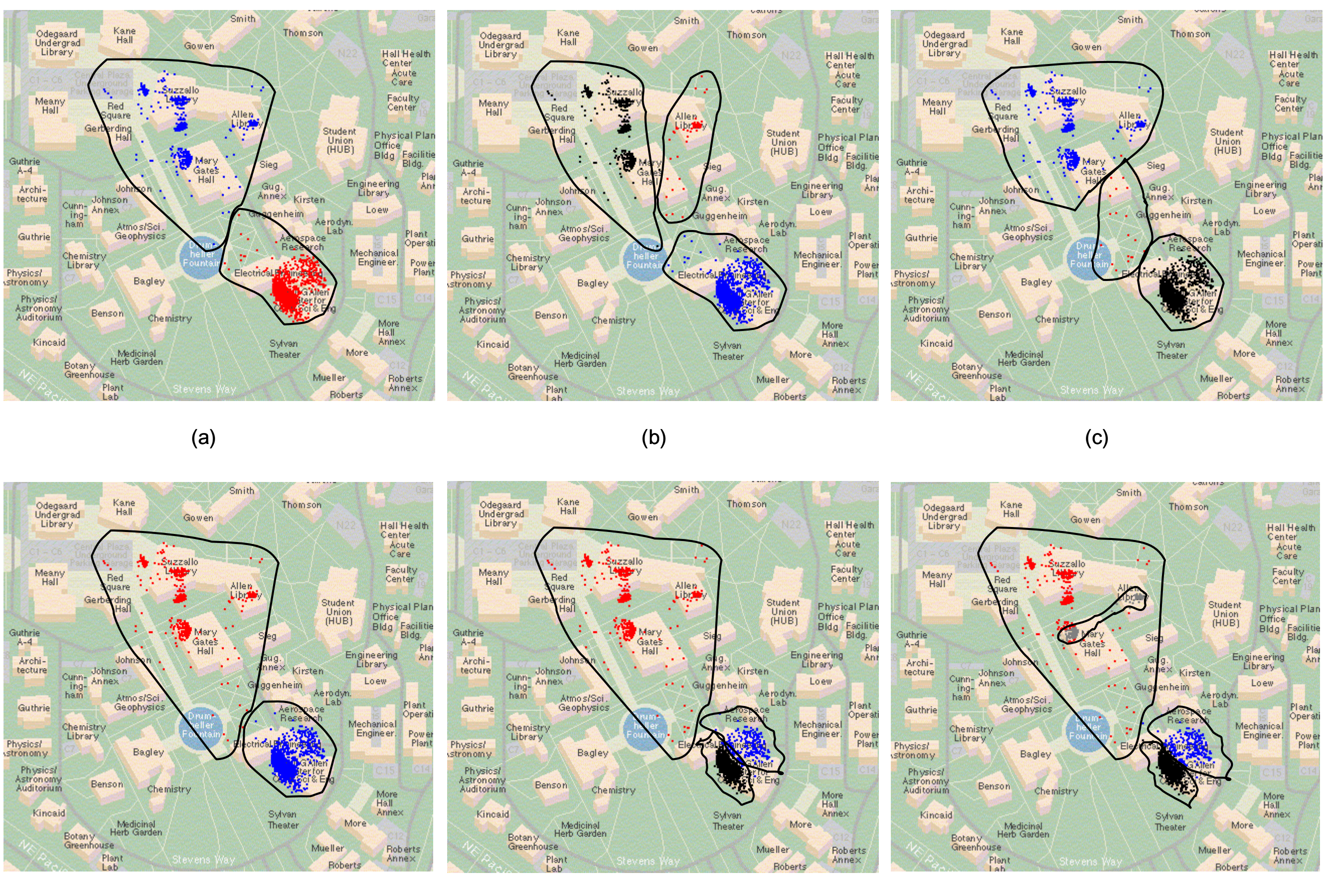


Figure 2. The clustering results from k-means and GMM. (a), (b), and (c) are the results from k-means. (d), (e), and (f) are from GMM. The number of clusters is set to 2 for (a) and (d), 3 for (b) and (e), and 4 for (c) and (f). In (c), the number of clusters is set to 4, but k-means could only find an optimal clustering with 3 clusters. Note the interesting cluster-within-a-cluster in (f).

3.3 Time-based Clustering

To overcome the drawbacks of existing clustering algorithms, an appropriate algorithm should be able to eliminate the intermediate locations between important places, and determine the number of clusters (important places) autonomously. Also, it should be simple enough to run on a simple mobile device as a background task.

The basic idea of our approach is to cluster the locations along the time axis. As a new location measurement is reported, the new location is compared with previous locations. If the new location is moving away from previous locations, the new location is considered to belong to a different cluster than the one for the previous locations. Figure 3 illustrates our approach. Suppose that the user moves from place A to place B. While the user is at place A, the location measurements are all close together (within a certain distance of each other – a parameter of our algorithm) and considered to belong to one cluster, namely, cluster *a*. As the user moves toward place B, the location measurements move away from cluster *a*. On the way to place B, a few small intermediate clusters are generated (*i1, i2*, and *i3*). And, when the user gets to place B and stays there for a while, a new cluster (cluster *b*) is formed. If a cluster's time duration is longer than a threshold (the second parameter of our algorithm), the cluster is considered to be a significant place. In the figure, cluster *a* and cluster *b* are determined to be the significant places while the other clusters in between are ignored.

The algorithm is depicted in Table 1 (*d* and *t* are our distance and time threshold parameters). When a new location measurement event is generated by Place Lab, the **cluster** function is invoked. The current cluster *cl* is the set of location measurements that belong to the current cluster. The pending location *ploc* is used to eliminate outliers. Even if the new location is far away from the current cluster (distance is larger than the distance threshold *d*), the algorithm does not start a new cluster right away with the new location. Instead, the algorithm waits for the next location to determine if the user is really moving away from the cluster or the location reading was just a spurious outlier. The *Places* contain the significant places where the user stays longer than the time threshold *t*.

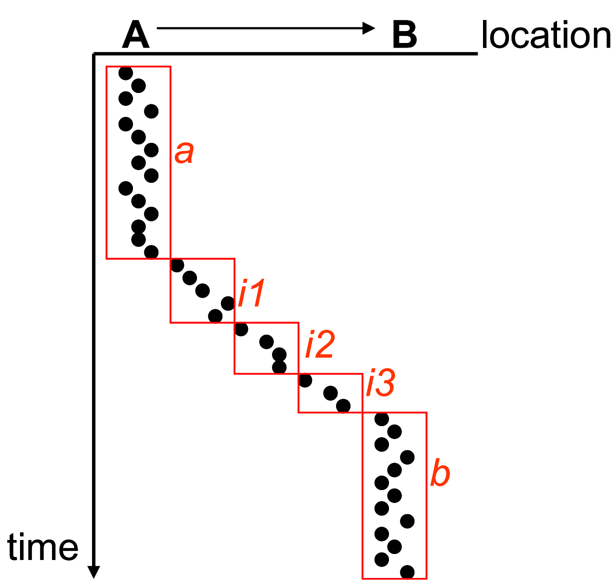


Figure 3. The illustration of the time-based clustering algorithm.

When a new location measurement is generated from Place Lab, the algorithm compares the distance between the mean position of the current cluster and the new location with the distance threshold *d*. If the distance is less than *d*, the new location is added to the current cluster and the pending location is set to null (lines 2-3). If the distance is larger than *d*, the algorithm checks if there is a pending location (line 5). If there is a pending location, the algorithm closes the current cluster and checks the time duration of the current cluster (the difference between the oldest and newest locations in the cluster). If the time duration of the cluster is longer than the time threshold *t*, the cluster is added to the significant places (lines 6-7). Then, the algorithm starts a new cluster with the pending location and checks if the new location can be in the same cluster as the pending location (lines 8-14). If the distance between the new location and the current cluster is larger than *d* but there is no pending location, the algorithm set the pending location to the new location (line 16).

When a cluster is added to the set of significant places, the algorithm checks if the cluster is the same as one of the existing clusters (their centroids are within distance *d*/3 of each other). In order to identify more fine-grain places, we use a smaller threshold (*d*/3) than the one used for forming clusters (*d*). The smaller threshold works because the difference between the averages of the location measurements over a period of time is likely to be much smaller than the difference between individual location measurements. If the newly added cluster is close enough to one of the clusters, then the two clusters are merged.

Table 1. Time-based clustering algorithm

```
cluster(loc)
input: measured location loc
state: current cluster cl,
       pending location ploc,
       significant places Places

 1: if distance(cl, loc) < d then
 2:   add loc to cl
 3:   ploc = null
 4: else
 5:   if ploc != null then
 6:     if duration(cl) > t then
 7:       add cl to Places
 8:     clear cl
 9:     add ploc to cl
10:     if distance(cl, loc) < d then
11:       add loc to c
12:       ploc = null
13:     else
14:       ploc = loc
15:   else
16:     ploc = loc
```

Unlike the other clustering algorithms that require all the location measurements to compute clusters, our algorithm computes the clusters incrementally as the new location measurements come in. Therefore, the significant locations can be extracted at run-time and the computation is simple enough – comparing the distance between the new location and the current cluster – to be easily supported on small battery-powered mobile devices.

Figure 4 shows the clusters generated by our algorithm. It generates four clusters from the trace. Each cluster corresponds to one of the places the user has visited. The intermediate locations between these significant places are ignored and the individual locations that make up the clusters corresponding to the significant places are the only ones shown in the figure. One interesting observation is that the shape of the bottom-right cluster is quite different from what we started with in the raw data. In the raw data, the locations in that place are clustered into two groups. But, the cluster generated by the algorithm includes only the locations in the group to the bottom-left. The user (author) was staying at the same place inside the building, but the location estimate from Place Lab was intermittent due to the variations of the signals from access points. It was moving back and forth between the two groups. It stayed in the bottom-left group for a period of time longer than the time threshold, and moved to the upper-right group for time periods shorter than the time threshold. Therefore, the upper–right group of locations were dropped by the time threshold.

The number of clusters and the size of each cluster depend on the two parameters: *d* and *t*. The distance threshold determines the size of the clusters. If a new location is further than *d* away from the mean of the current cluster, the new location is considered to belong to a new cluster. Thus, if *d* is smaller than the variations of the measurement, the algorithm may miss some significant places with high variations. In such places, even if the user stays in the same place, the location measurement can often vary a lot and the algorithm starts a new cluster accordingly. In this case, the algorithm generates several fragmented clusters with short time duration instead of one cluster with long time duration. These fragmented clusters with short time duration are filtered out by the time threshold. If *d* is too large, the size of clusters may become large and absorb the intermediate locations. Also, small adjacent clusters may be collapsed into one big cluster.

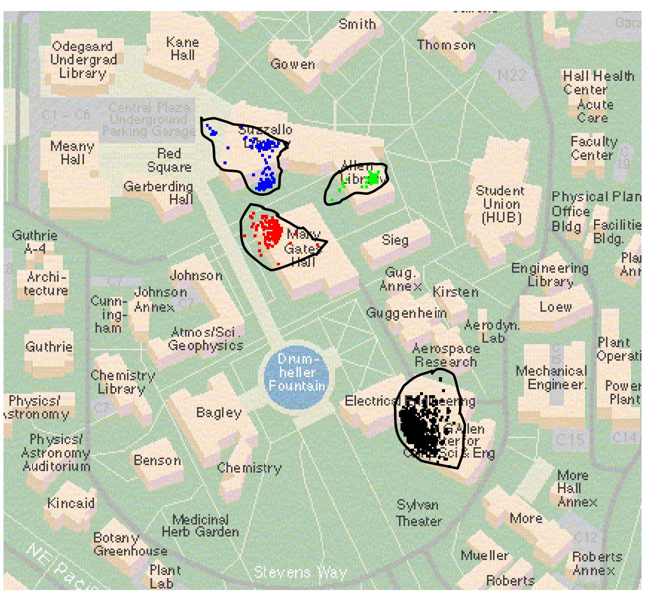


Figure 4. The significant places extracted by the time-based clustering algorithm.

The time threshold *t* determines the number of significant places. Only the clusters with longer time duration than *t* are added to the set of significant places. If *t* is too small, some unimportant clusters can be included in the set of significant places. On the other hand, if *t* is too large, we may miss some significant places that the user stops at for a time less than *t*.

The graphs in Figure 5 show the number of significant places found for different time and distance thresholds for two different traces of one of the authors' daily life. In both traces, there appears to be a noticeable knee in the curve between 20 and 30m. Below 20m, there are too many short duration clusters generated. Above 30m, the resulting number of clusters is quite stable. Thus, we choose the distance threshold value to be between 30m and 50m. Eventually, we will want to determine this value automatically as well and believe we will be able to do so with a few days of training data.

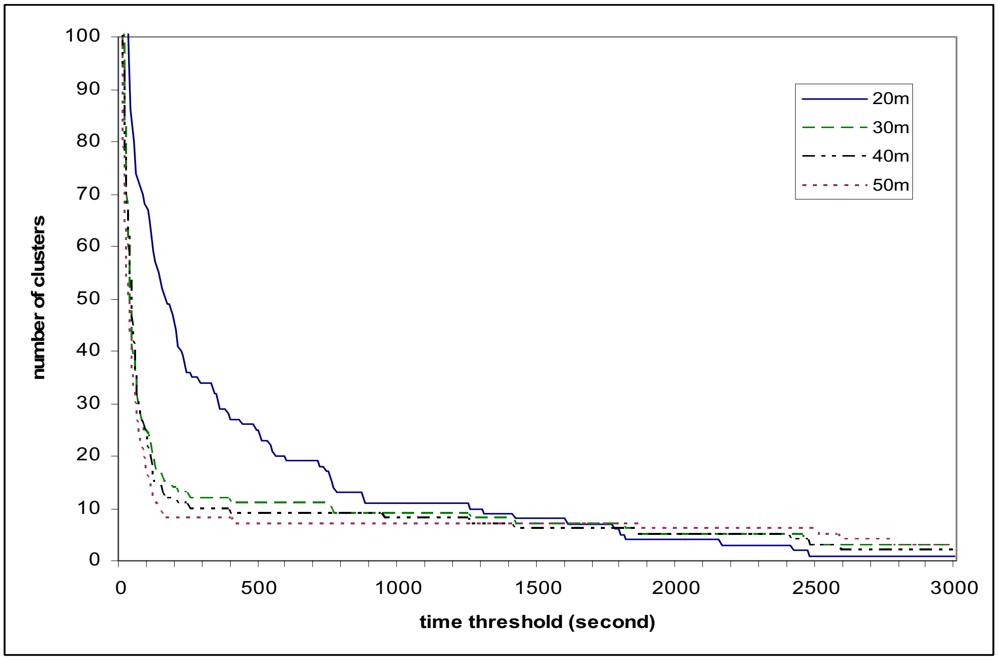
For the time threshold, for both traces, the graph becomes flat when the value of *t* is longer than about 300 seconds (for values of d > 30m). If the time threshold is shorter than 300 seconds, some unimportant clusters can be chosen as significant places. Therefore, we chose the time threshold to be at least 300 seconds. Depending on the user's preference, the time threshold can be set to a larger value. Ashbrook and Starner [1] did not have a clear knee in their curve and chose a value of 600 seconds somewhat arbitrarily. We believe the reason we have a more pronounced knee is the more continuous nature of our location readings in both indoor and outdoor settings.

4. EXPERIMENTAL EVALUATION

To evaluate the effectiveness of the proposed place extraction scheme, we must first decide on a criterion for evaluation. Intuitively, a place extraction scheme should be judged on how well it identifies the locales that a user deems important. The goodness of our results then, can be measured both in terms of *accuracy*, the lack of erroneously identified places, and *completeness*, the fraction of all places correctly identified. To this end, we run our time-based clustering algorithm on location traces of users' daily activity, and assess the results with both user-composed logs of the places visited (or "place logs") and map visualizations as ground truth.

4.1 Trace Collection

As noted in section 3.1, our location traces were collected using Place Lab. We used Place Lab's simple "centroid tracker", taking samples and logging location once per second. Traces were collected with wireless mobile devices during the daily activities of the first and second authors – corresponding activity logs were also composed at the end of the day. As the two authors typically stay within the Seattle city limits, and as most of this area is covered by the Place Lab AP database, there were no problems with location data being unavailable.



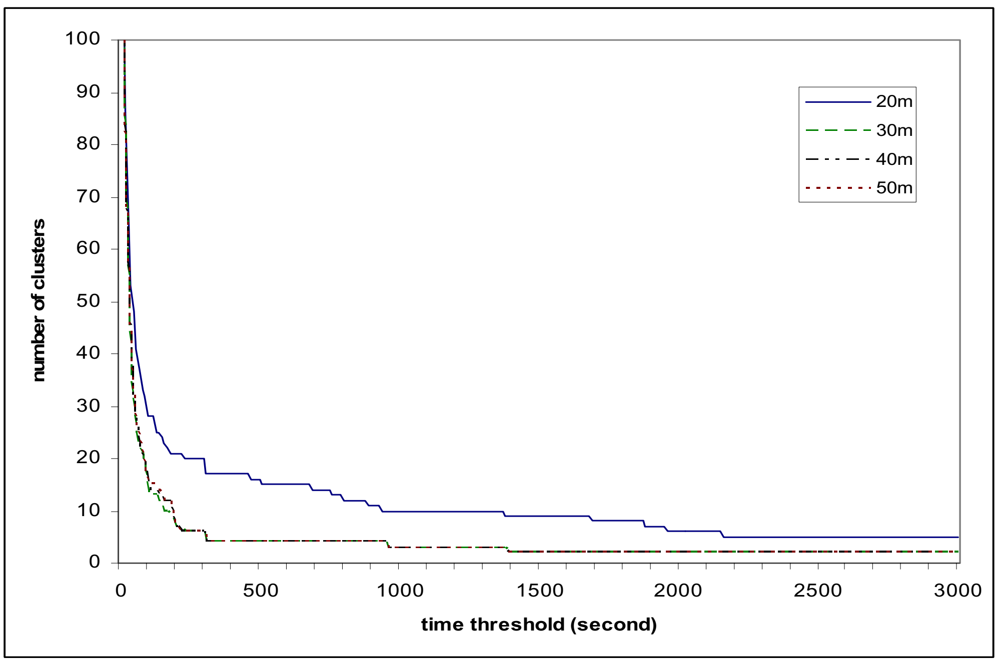


Figure 5. The number of significant places found for different distance and time thresholds.

For the purpose of evaluation, we chose one representative trace from each person. The first trace, over a small area, is of an author's daily errands around the university campus with a total duration of about 2 hours. The trace log was started when the user was in his office, place 1 in Figure 6(a). After about 10 minutes in his office, the author left to go home. On his way off campus, the author ran errands in five buildings across campus (places 2 through 6), staying 9 to 20 minutes in each place.

The second trace is of an author's daily movement between home, work, lunch, school, and a friend's house with a total duration of about 12 hours. The trace starts at the author's home (place 1 in Figure 7(a)) in the morning. After about 30 minutes, he headed to his place of work (place 2). At work, he attended a meeting in a conference room in one corner of the building, and spent the rest of the time at his desk in the other corner. After a few hours, he left to attend two meetings in a building on campus (place 3) – each meeting was held in a different room. After the second

Table 2. Detailed description of places visited in the first trace and the duration of stay in each place.

Place and Duration	Description of the Place
place 1: (10 min)	Indoor: 3rd floor office in a building; has windows; APs internal and external to the building are visible.
place 2: (17 min)	Indoor: Lobby of adjacent building; mostly concrete; internal APs of both buildings are visible.
place 3: (9 min)	Indoor: In the 3-story high atrium of a building; APs internal to building are visible.
place 4: (15 min)	Indoor: In the middle of a 1st floor corridor of a building; APs internal to building are visible.
place 5: (20 min)	Outdoor: On a bench between two buildings; open but with trees; APs from both buildings visible
place 6: (14 min)	Outdoor: On stair between two buildings; narrow "alley"; APs from near and distant buildings are visible

Table 3. Detailed description of places visited in the second race and the duration of stay in each place.

Place and Duration	Description of the Place
place 1: (35 min)	Indoor: 2nd floor apartment; has windows; APs internal and external to apartment building are visible.
place 2: (8 hour 20 min)	Indoor: 6th floor of building; office and conference room; APs internal and external to building are visible.
place 3: (45 min)	Indoor: 5th floor of campus building; offices on east and west sides; APs internal to building are visible.
place 4: (45 min)	Indoor: At table in open-air restaurant; APs external to building are visible.
place 5: (1 hour 40 min)	Indoor/Outdoor: At outdoor shopping mall; APs in various nearby buildings are visible.
place 6: (7 min)	Outdoor: On rooftop patio of apartment building; APs from both near and distant buildings are visible.

meeting ended, he returned to his place of work. At lunch time, he went out to eat at a restaurant a few blocks away (place 4). At the end of the day, he visited a shopping mall (place 5) and his friend's house (place 6) before returning home.

We evaluate our place extraction algorithm on these two traces and present the results below. For a more detailed description of each visited place, including environmental characteristics that might affect WiFi signal, please see Tables 2 and 3.

4.2 Experimental Results

Visualizations of the raw trace data for the first and second traces are shown in Figures 6(a) and 7(a) respectively. These figures also show the places listed in each author's place log as circles labeled with a number. Figures 6(b)-(d) and 7(b)-(c) show the results of time-based clustering applied to the traces for various values of d and t. The results depicted in these figures are evaluated in terms of accuracy and completeness below.

Accuracy

The raw traces shown in Figures 6(a) and 7(a) show a large number of trace points scattered along what are obviously routes between places (e.g. sidewalks, roads). We can see in Figures 6(b)-(d) and 7(b)-(c) that for each trace and each pair of clustering parameters, the scattered points between places have been excluded from the final result. Furthermore, a comparison of the results for each trace with the authors' place logs shows that each extracted place does actually correspond to a visited place.

Completeness

The results for the first and second traces show that the completeness of a set of extracted places depends largely on the choice of parameters d and t.

For the first trace, Figure 6(b) shows the places extracted when d = 30 meters and t = 300 seconds; note that only five of the six author-identified places were found in this case. The raw data in Figure 6(a) shows that the trace points around the missing place (place 6) are scattered over a relatively wide area, this is probably due to a high variation in the set of visible APs. Thus, by increasing d to 50 meters (figure 6(d)) we can compensate for scattered location estimates and recover place 6. Alternatively, if we hold d = 30 meters and increase t to 600 seconds (Figure 6(c)), we lose places 1 and 3. This is because the author spent less than 10 minutes on errands in places 1 and 3.

In the second trace, all major places (and some sub-places) with the exception of place 6 were identified with d = 50 (d = 30 yields the same result). Similar to the case of a missing place 6 in trace 1, it is likely that place 6 in trace 2 was not extracted because the surrounding trace points were scattered by occasionally visible, distant APs. This problem is likely to be lessened as Place Lab evolves to include more sophisticated tracking and AP placement schemes. In the meantime, this problem could be avoided by using a larger d value, and by using more trace data (which would presumably include more time spent in the same place, and so give the wide area enough "weight" to be considered a place).

It is also interesting to note that our algorithm was able to make the distinction between places 1 and 2 in the first trace with d set to either 30 or 50 meters. This is surprising because in the raw trace data these places look like the same cluster. Similarly, various "sub-places" could be identified depending on the value of the d parameter. For example, in the second trace the conference room and office at place 2 could be distinguished with d = 30 or 50 meters, as could the two offices in place 3 with d = 30 meters. The latter observations support the intuitive notion that a smaller *d* value will increase the chance that sub-places are extracted (At the same time, it will also increase the chance to miss the places with a high variation in visible APs).

5. CONCLUSIONS AND FUTURE WORK

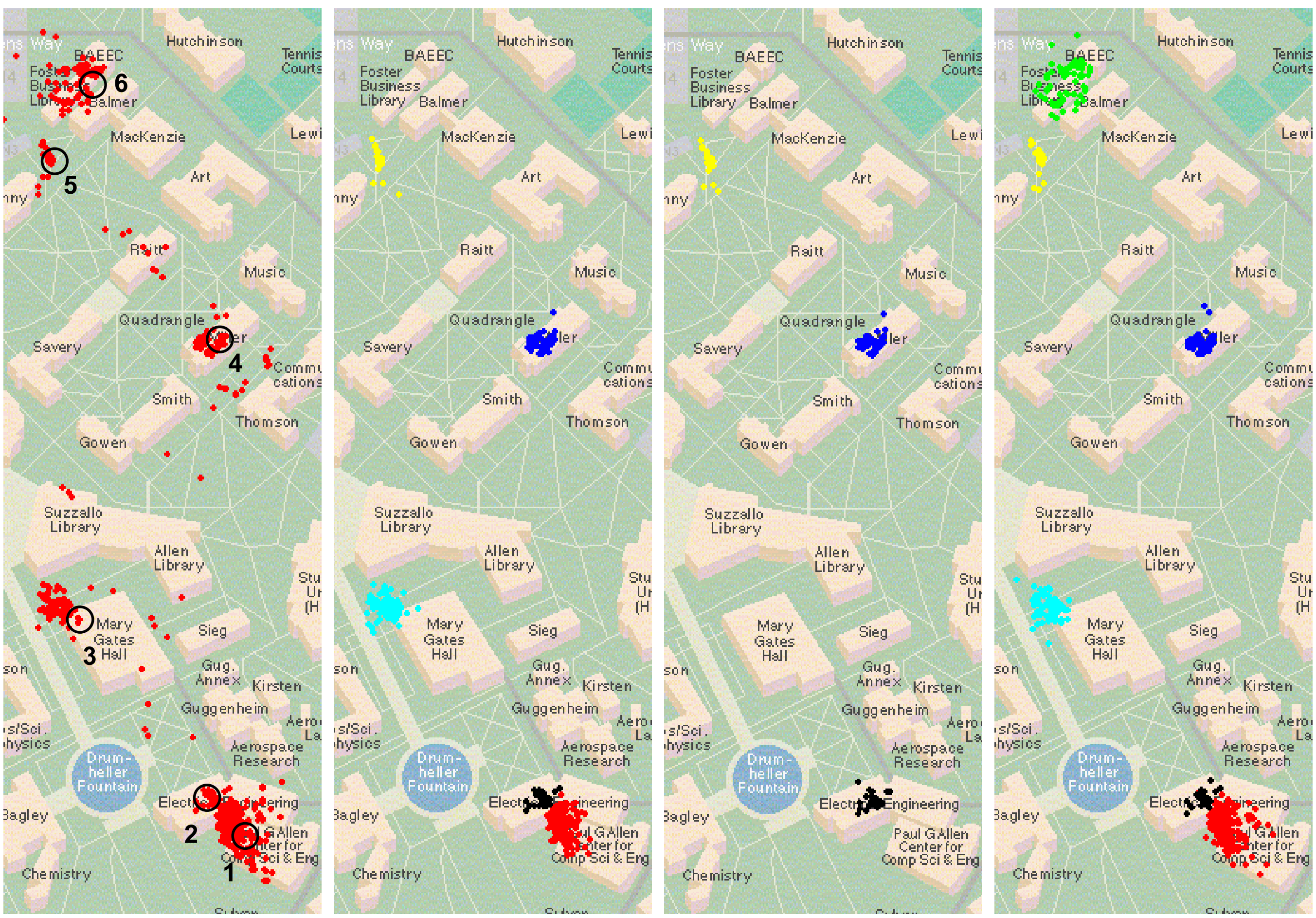
In this paper, we presented an algorithm for extracting significant places from a trace of coordinates. The significant places where the user spends considerable amount of time appear as clusters of locations in the trace. Although this is basically a clustering problem, the popular clustering algorithms are not quite right for this particular problem for three reasons: (1) the number of clusters is an a priori parameter; (2) the generated clusters include unimportant locations; and (3) the clustering algorithms require a significant amount of computation.

Our simple algorithm clusters the locations along the time axis and extracts the clusters without a priori knowledge of the number of clusters. In addition, the clusters generated by our algorithm are more likely to tightly bound these places and to exclude extraneous coordinates. We also showed how we determined two key parameters of our clustering approach (the distance and time threshold, *d* and *t*) which we are working on learning automatically.

We also evaluated our algorithm with real trace data collected using Place Lab [14], a coordinate-based location system that uses a database of locations for WiFi hotspots. Our initial experimental results show that our algorithm extracts the most significant places successfully.

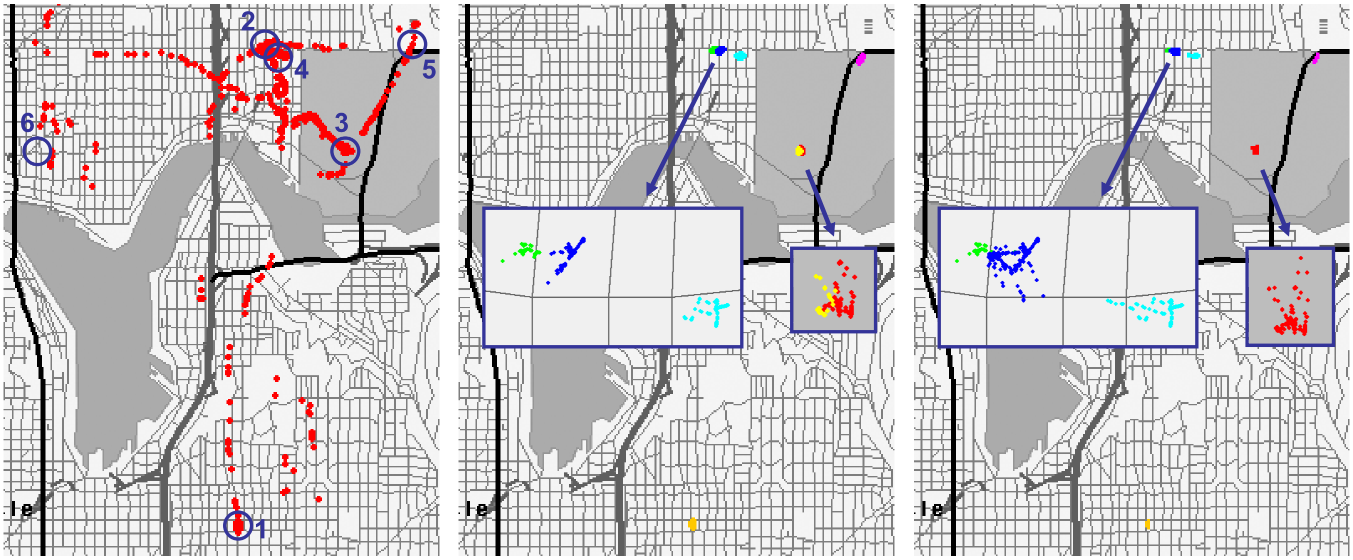
The extracted places need to be labeled in order to have semantic meanings and to be used by other applications. We are working on automatic labeling of the extracted places using additional information such as the user's calendar and other users' labeling.

Another direction for future work is to predict a user's destination from their current location and past observations of their movements. With a slight modification, our algorithm can record the arrival and leaving time to and from the extracted places. For example, each place can have the information on what time of day, or which day of the week the user visited that place. With this information, we can better predict users' destinations as they go about their day and provide proactive assistance [12].



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

Figure 6. Visualization of the campus scale trace. (a) shows the raw trace data. (b) shows the significant places extracted when *d* = 30m and *t* = 300sec. (c) is when *d* = 30m and *t* = 600sec. (d) is when *d* = 50m and *t* = 300sec.



(a) (b) (c)

Figure 7. Visualization of the city scale trace. (a) shows the raw trace data. (b) shows the significant places extracted when *d* = 30m and *t* = 300sec. (c) is when *d* = 50m and *t* = 300sec. When *d* = 30m, two sub-places are extracted from place 3.

6. REFERENCES

[1] Daniel Ashbrook, Thad Starner. Using GPS to learn significant locations and predict movement across multiple users. In Personal and Ubiquitous Computing, Volume 7, Number 5, October 2003.

[2] Jeffrey D. Banfield, Adrian E. Raftery. Model-based Gaussian and Non-Gaussian Clustering. Biometrics 49, September 1993.

[3] Gaetano Borriello, Waylon Brunette, Matthew Hall, Carl Hartung, Cameron Tangney. Reminding about Tagged Objects using Passive RFIDs. To appear in Ubicomp 2004.

[4] Richard O. Duda, Peter E. Hart. Pattern Classification and Scene Analysis. John Wiley & Sons, 1973.

[5] Ramaswamy Hariharan, Kentaro Toyama. Project Lachesis: Parsing and Modeling Location Histories. To appear in the 3rd International Conference on Geographic Information Science, October 2004.

[6] John Krumm, Ken Hinckley. The NearMe Wireless Proximity Server. To appear in Ubicomp 2004.

[7] Kari Laasonen, Mika Raento, and Hannu Toivonen. Adaptive On-Device Location Recognition. In Proceedings of the 2nd International Conference on Pervasive Computing (Pervasive 2004), Vienna, Austria, April 2004.

[8] Lin Liao, Dieter Fox, and Henry Kautz. Learning and Inferring Transportation Routines. In Proceedings of AAAI-04 , 2004.

[9] Matalia Marmasse, Chris Schmandt. Location-aware information delivery with comMotion. In Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing (HUC 200), Bristol, UK, September 2000.

[10] Donald J. Patterson, Oren Etzioni, Henry Kautz. The Activity Compass. In Proceedings of the 1st International Workshop on Ubiquitous Computing for Cognitive Aids (UbiCog '02), September 2002.

[11] Donald J. Patterson, Lin Liao, Dieter Fox, Henry Kautz. Inferring High-Level Behavior from Low-Level Sensors. In Proceedings of the 5th International Conference on Ubiquitous Computing (Ubicomp 2003), Seattle, WA, October 2003.

[12] Donald J. Patterson, Lin Liao, Kryzstof Gajos, Michael Collier, Nik Livic, Katherine Olson, Shiaokai Wang, Dieter Fox, Henry Kautz. Opportunity Knocks: a System to Provide Cognitive Assistance with Transportation Services. To appear in Ubicomp 2004.

[13] Dau Pelleg, Andrew Moore. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In Proceedings of the 17th International Conference on Machine Learning, 2000.

[14] Bill Schilit, Anthony LaMarca, Gaetano Borriello, William Griswold, David McDonald, Edward Lazowska, Anand Balachandran, Jason Hong and Vaughn Iverson. Challenge: Ubiquitous Location-Aware Computing and the Place Lab Initiative. In Proceedings of the 1st ACM International Workshop on Wireless Mobile Applications and Services on WLAN (WMASH 2003), San Diego, CA, September 2003.