Platys: User-Centric Place Recognition

Chung-Wei Hang, Pradeep K. Murukannaiah, and Munindar P. Singh
North Carolina State University
chang@ncsu.edu, pmuruka@ncsu.edu, m.singh@ieee.org

Abstract

Emerging mobile applications rely upon knowing a user's location. A (geospatial) position is a low-level conception of location. A *place* is a high-level, user-centric conception of location that corresponds to a well-delineated set of positions. *Place recognition* deals with how to identify a place. Traditional place-recognition approaches (1) presuppose manual tuning of place parameters; (2) limit themselves to specific sensors; or (3) require frequent power-consuming sensor readings. We propose Platys, an adaptive, semisupervised approach for place recognition, which makes weak assumptions about place parameters, and the types and frequencies of sensor readings available. We evaluate Platys via a study of six users. A comparison with two traditional approaches indicates that Platys (without parameter tuning) performs better than traditional approaches (with optimally tuned parameters).

Introduction

To adapt to the changing location of a user is a distinguishing feature of mobile applications. A position refers to the physical location of a user—absolute (e.g., geographical coordinates) or relative (to an object of known position).

Unlike a position, a place is a conceptually well-delineated set of (not necessarily contiguous) positions. Thus a place describes a user's context better than a position and provides a high-level abstraction for applications. For example, a user may wish his automated cell-phone ringer to remain silent in a *restaurant* and loud at a *supermarket*, wherever each might be.

The objective of place recognition is to identify a set of positions of a user that would ordinarily be treated uniformly by an application. Often, staypoint approaches are used for place recognition (Hariharan and Toyama 2004; Zheng et al. 2011). A *staypoint* is a set of positions within a specified *radius* (say, 200 m) that a user visits for a specified *duration* (say, 30 minutes). Staypoints help retain "relevant" positions and filter out irrelevant ones (e.g., while traveling). From staypoints, probabilistic approaches (Hariharan and Toyama 2004) and clustering (Zheng et al. 2011) help extract places. These approaches presume that radius and duration are tuned. Staypoint approaches suffer from three limitations, which we address.

First, no fixed parameter values are appropriate for all places. Consider a user Alex with two places of interest (1) *daughter's school*, for pick up and drop off, which is used in an application that notifies his daughter when he is near the school; (2) *jogging trail* around a lake, which is used in an application to track time spent exercising. These applications need different radii and durations: *daughter's school* $\langle 50\ \mathrm{m}$, two minutes$\rangle$ and *jogging trail* $\langle 2.5\ \mathrm{km}$, one hour$\rangle$.

Second, places are *user-centric*. For example, the parameters for *school* for a student $\langle 500\ \mathrm{m}$, seven hours$\rangle$ differ from *school* for a parent $\langle 50\ \mathrm{m}$, two minutes$\rangle$. It is nontrivial to identify such distinctions in an unsupervised manner. Further, unsupervised approaches may eventually require a symbolic name for the user to understand the place.

Third, place recognition requires multiple data sources. For example, GPS outdoors, WiFi and Bluetooth indoors, user input, and so on. Data collection from these sources is often expensive (e.g., sensors cost energy) or intrusive.

We describe Platys, a semisupervised place-recognition approach. Platys takes as input (1) unaligned, intermittent readings at different *timepoints* from multiple sensors and web services and (2) user-provided labels for places of interest, a few times per place. Some of the readings are those infrequently collected by Platys. Others, especially expensive GPS readings, are saved whenever *another* application happens to save. Thus the readings are unaligned with each other. Platys learns the parameters of places and classifies any timepoint as belonging to a particular place.

Platys treats the information collected closest to each timepoint as its features. For example, in Figure 1, 18:00 has four features: a GPS reading at 18:05, one WiFi scan at 17:27 and another at 18:02, and a Bluetooth scan at 16:52. Platys weighs these features based on how long before or after the selected timepoint they occur. Then, it measures the similarity between the selected timepoint and a timepoint labeled as a place (e.g., 17:27 labeled as *Lab* and 20:12 labeled as *Piccola Italia*) to determine which place a given timepoint is likely to belong to. For example, 18:00 is more likely to be *Lab* than *Piccola Italia*.

Platys does not predefine the granularity of places. Instead, it extracts places by learning the *similarity boundary* between places (for each user). Platys assumes that a user

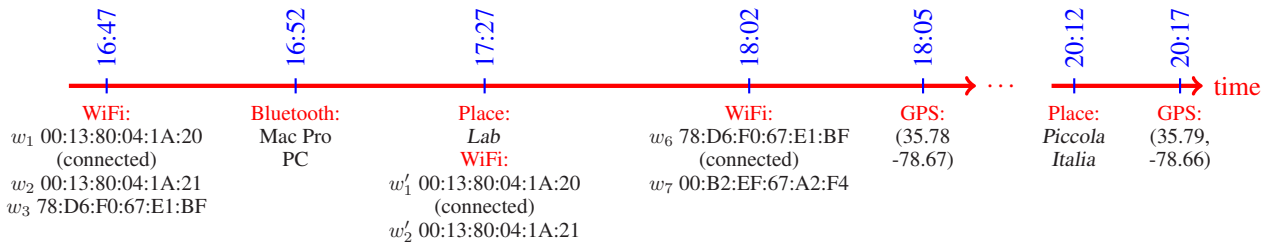

Figure 1: Unaligned sensor data and user labels collected by Platys. The features of timepoint 17:27 are relevant for place *Lab*. The closer a feature is to a timepoint the more relevant it is.

visits important places sufficiently often and makes weak assumptions on the types and frequencies of the sensor data, and the similarity metrics.

Approach

Place-recognition is functionality provided by the *Platys* middleware (Murukannaiah and Singh 2012a), which runs as a background service on Android phones, gathering data from available sensors (here, GPS, WiFi, and Bluetooth). Platys provides an application using which a user can (a) change sensor configurations, e.g., frequency of scanning, and (b) label the current or a recently visited place. The user may label a place any time; Platys prompts the user at random intervals.

Problem Formalization

We wish to determine a user's place at any given time, based on unaligned historical sensor data and user labels. For example, in Figure 1, what is the user's place at 17:35?

Platys collects the following timestamped information for each user.

- **GPS:** a series of GPS (latitude-longitude) coordinates $G = \{g_1, \ldots, g_{|G|}\}$.
- **WiFi:** a series of WiFi signals $W = \{w_1, \ldots, w_{|W|}\}$, each containing an identifier (MAC address) and RSSI (received signal strength indicator).
- **Bluetooth:** a series of Bluetooth signals $B = \{b_1, \ldots, b_{|B|}\}$, each containing an identifier (MAC address) and RSSI.
- **Points of interest (POI):** a series of Google Places[1] $GP = \{gp_1, \ldots, gp_{|G|}\}$, each $gp_i = \{poi_{ij}\}$ being a set of POIs based on $g_i \in G$.
- **User labels:** a series of labels $P = \{p_1, \ldots, p_{|P|}\}$.

Now the technical problem is: Given G, W, B, GP, and P of a user, how can we determine (1) whether at time x the user is located at one of the labeled places $p_i \in P$; and (2) if so, which place?

Given a timepoint, our approach is to determine

- **Features** of the timepoint from the closest sensor readings.
- **Similarity** of the timepoint's features to features of known places.

[1] https://developers.google.com/places/

Relevant Features

The *features* $F(x)$ of a timepoint x are the closest prior and subsequent readings from each of GPS, WiFi, Bluetooth, and POI data sources. Below, $t(f)$ refers to the time at which reading f is taken.

One reading may contain multiple data points. For example, a WiFi scan may contain multiple access points—each becomes a feature. We assign a *relevance measure* to each feature $f \in F(x)$ based on the time difference $\Delta t = |t(f) - x|$ (in milliseconds), using the logistic function:

$$r(f) = \frac{1}{1 + e^{\alpha \times \Delta t - \beta}}.$$

We set $\alpha = 10^5$ and $\beta = 6$ so that readings more than two hours apart have little bearing on each other. We choose these user-and place-independent parameters based on our observations of users.

We define the features $F(p)$ for each place p as the features of timepoints where it is labeled. Platys assigns timepoint x to p if $F(x)$ and $F(p)$ are sufficiently similar; otherwise, it leaves x as belonging to no place.

Next we describe how to measure similarity and determine a similarity threshold for each place.

Similarity Metrics

We set the similarity between different types of features (e.g., WiFi and GPS) to zero, and measure the similarity between same type of features as follows.

GPS The similarity of two GPS coordinates g_i and g_j relates inversely to their Euclidean distance $d(g_i, g_j)$ (in km).

$$sim_{\text{GPS}}(g_i, g_j) = \frac{1}{1 + e^{50 \times d(g_i, g_j) - 6}}$$

For example, consider three GPS coordinates shown in Figure 3.

$$sim_{\text{GPS}}(g_X, g_Y) = 0.0 \text{ (distance: 1057 m), and}$$
$$sim_{\text{GPS}}(g_X, g_W) = 0.1 \text{ (distance: 211 m).}$$

WiFi and Bluetooth The similarity of two WiFi signals with different identifiers (MAC address) is zero. If the identifiers match and the user connects to the access point on both timepoints, the similarity is one. Otherwise, we scale the difference in their RSSIs to $[0, 1]$ and take similarity as one minus the resulting value.

For example, in Figure 1, suppose the WiFi scan at 16:47 is $W = \{w_1, w_2, w_3\}$ and the scan at 17:27 is $W = \{w_1, w_2, w_4\}$. If the user connects to one of the common w_i, $sim_{\text{WIFI}}(W, W') = 1$. Otherwise, the similarity depends on which of the common signals have the closest RSSI values. Suppose that is w_2 with RSSI of -51 and -49, respectively. Then, $sim_{\text{WIFI}}(W, W') = 0.98$.

The similarity between two Bluetooth scans $sim_{\text{BLUETOOTH}}(B, B')$ is calculated similarly.

Points of Interest The Google Place API returns a Google Place containing one or more potentially overlapping POIs related to a GPS coordinate. For example, a GPS coordinate can belong to both Lake Johnson and Southwest Raleigh, because Lake Johnson is in Southwest Raleigh.

We adapt Lin's (Lin 1998) approach to measure the similarity of Google Places based on the frequency of a POI. Intuitively, matching a rarer POI (e.g., McDonald's) is more valuable than matching a more frequent POI (e.g., Raleigh). Below, n_{poi} is the number of occurrences of a POI and $Prob(poi) = \frac{n_{poi}}{\sum_{p \in gp} n_p}$ is the probability of visiting a POI.

Thus, the similarity between two Google Place features $sim_{POI}(gp_i, gp_j)$ is

$$sim_{POI}(gp_i, gp_j) = \frac{2 \times I(gp_i \cap gp_j)}{I(gp_i) + I(gp_j)}.$$

where $I(gp) = -\sum_{poi \in gp} \log Prob(poi)$.

Continuing our example from Figure 3, consider these POIs, their matched positions, and their probabilities of being visited:

Crab Orchard Rd X; 0.00024
Gorman St Z; 0.00024
Avent Ferry Rd Y; 0.0025
Lake Johnson Nature Park (LJNP) X, Y; 0.025
Southwest Raleigh X, Y, Z; 0.12

Interestingly, $sim_{POI}(gp_X, gp_Y) = 0.4451 > sim_{POI}(gp_X, gp_Z) = 0.1711$, although $sim_{GPS}(X, Y) < sim_{GPS}(X, Z)$. What matters here is that gp_X and gp_Y match on the POI *LJNP*, which has a lower probability of being visited than *Southwest Raleigh*—the smallest POI common to gp_X and gp_Z. In reality too, g_X and g_Y are within LJNP, whereas g_Z is a home that happens to be nearby. If we used only GPS data, we would draw an erroneous conclusion.

Timepoint Similarity The overall similarity between two timepoints x and y is the maximum similarity based on any of the above features.

Similarity Boundary

We classify a timepoint x as place p based on $Prob(p|x)$: the probability of the user being in place p given the features of timepoint x. Following Bayes' rule,

$$Prob(p|x) \propto Prob(x|p)Prob(p).$$

We assume $Prob(x|p) \propto sim(F(x), F(t(p)))$. Assuming a uniform prior, $Prob(p)$, we classify x to the place $\hat{p}$ where $Prob(\hat{p}|t)$ is the highest among all p.

However, many timepoints should belong to no place. We adopt Expectation Maximization (EM) (Dempster, Laird, and Rubin 1977) to find an appropriate *similarity boundary* for each place. A similarity boundary groups mutually similar timepoints. Our approach begins with a fairly large similarity boundary (with similarity, $\epsilon = 0.5$); iteratively clusters a set of timepoints; and reduces the similarity boundary (i.e., increases ϵ) based on the mean similarity (ϵ') of the current clustered timepoints until the boundary converges. Using a large initial similarity boundary helps us quickly filter out timepoints that are unlikely to be assigned to a place. Then, for each place p, repeat these steps until convergence:

E-step:

- Compute $\mathsf{timepoints}(p) = \{x | Prob(p|x) \geq \epsilon\}$. These are all the timepoints sufficiently similar to p.

M-step:

- Calculate the new similarity boundary as the mean similarity of the timepoints in p:

$$\epsilon' = \frac{\Sigma_{x \in \mathsf{timepoints}(p)} Prob(p|x)}{|\mathsf{timepoints}(p)|}$$

- Terminate if the boundary converges. Here, δ, a convergence parameter is 0.001.
  If $\epsilon' - \epsilon < \delta$, stop.
  Otherwise, set $\epsilon = \epsilon'$ and iterate.

Note that a place can include disjoint spatial regions. For example, a user may label all hotels as *hotel*. This is as it should be since in the user's context the fact that they are hotels is significant whereas their positions are not. For disjoint regions, Platys learns the subplaces individually and merges them by the label. When the user approaches any of the labeled hotels, Platys will identify them as the place *hotel*.

Evaluation

We compared Platys with two staypoint approaches (Hariharan and Toyama 2004; Zheng et al. 2011). These approaches are unsupervised—they take no inputs from users and only distinguish places (staypoints) from *nonplace*, which we treat as a distinct place. To make a fair comparison, we implemented a variant of Platys, *Place-or-not*, which only distinguishes places and nonplace. Additionally, our full approach, *Which-place*, identifies the specific place. Obviously, the effectiveness of *Place-or-not* is an upper bound on that of *Which-place*.

Data

No available datasets were adequate for our evaluation, so we created our own dataset based on six users. Each user carried an Android phone with the Platys middleware installed for two to 23 weeks. The Platys application collected a user's place labels (three–four times a day during normal waking hours plus whenever the user felt like labeling a place). It also recorded sensor data (including GPS, WiFi,

and Bluetooth)—depending on the phone model, other running applications, user settings, and available signals, from once in 10 seconds to once in several hours.

To enable our evaluation, each user provided place labels as ground truth for each timepoint where a GPS signal was logged for that user. To facilitate this task, we provided a web-based (large-screen) interface showing the timepoint as a pin on a map, relevant data from other sensors, and the user's trajectory around that time.

Our dataset contains a variety of users (one faculty member, one postdoc, and four graduate students from two departments), differing in their mode of transportation (drive, walk), mobility across states and countries, and frequencies of sensor data collection.

Evaluation Metrics

We measure the effectiveness of place-recognition approaches via the F-measure. We use *micro-averaging* so that significant places influence the F-measure more than insignificant places (in terms of their occurrence in the sampled timepoints) (Yang and Liu 1999). For example, if most of the timepoints in a sample belong to nonplace, micro-F-measure would be influenced more by how well an approach predicts the nonplace than other places.

Below, TP, TN, FP, FN refer to true and false positives and negatives. Then micro-averaged precision, recall, and F-measure are defined, for each user, as

$$\begin{aligned} \text{precision} &= \frac{\sum_{i=1}^{|P|} TP_i}{\sum_{i=1}^{|P|}(TP_i + FP_i)} \\ \text{recall} &= \frac{\sum_{i=1}^{|P|} TP_i}{\sum_{i=1}^{|P|}(TP_i + FN_i)} \\ \text{F-measure} &= \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}, \end{aligned}$$

Results

Figure 2 compares the F-measure of all four approaches. The micro-averaged F-measure, averaged across users, of both *Place-or-not* and *Which-place* reaches over 80%.

We varied the parameters of the two staypoint approaches from $\langle 3 \text{ minutes}, 20 \text{ m}\rangle$ to $\langle 1 \text{ day}, 96 \text{ km}\rangle$. The fixed parameters $\langle 30 \text{ minute}, 200 \text{ m}\rangle$ used by Zheng et al. (Zheng et al. 2011) are reasonable, though not optimal, for all users. With optimal parameters, the staypoint approaches might identify places more accurately than Platys. However, for most users, this advantage is negligible and, importantly, finding the optimal parameters is nontrivial. Further, a user can have places with different granularities. For example, User A has two peak levels of granularity around $\langle 25 \text{ minute}, 166 \text{ m}\rangle$ and $\langle 17 \text{ hour}, 700 \text{ m}\rangle$.

Which-place performs nearly as well as *Place-or-not* for all users except User C. Thus, once a timepoint is identified as a place, Platys can correctly identify the specific place most of the time. User C labeled places at a finer granularity than the data we collected can distinguish. For example, User C labeled *kitchen*, *garage*, *bedroom*, which had similar POIs and sensor data. Moreover, GPS coordinates are mostly unavailable indoors or may be erroneous (with error magnitudes in tens of meters, which approximate the sizes of the tagged places). Thus *Which-place* performs poorly for User C.

User D's phone collected GPS data at a much higher frequency than other users' phones (presumably due to other applications invoking GPS). Thus, both staypoint approaches (with appropriate parameters) and our approach identity and distinguish places accurately.

Figure 3 shows some example places identified for User A. Table 1 shows the precision, recall, and F-measure of these places in terms of the positions being classified. Our approach finds *Gym*, *Home*, and *Friend's place* accurately. For *Lake Johnson*, Platys finds a smaller region than the actual lake, the reason being that User A always tagged *Lake Johnson* at its entrance.

A limitation of our evaluation strategy is that we asked users to provide ground truth for GPS timepoints only. As a result, recognition accuracy seems low for places such as *Lab* and *Toxicology* where there is no GPS signal. *Which-place* mistakenly identifies the GPS timepoints scanned before the user enters or after he leaves *Lab* as *Lab* resulting a low recognition accuracy. In practice, Platys could correctly identify such places through a similarity boundary determined by WiFi and Bluetooth similarity. However, we don't have the ground truth (for timepoints with WiFi and Bluetooth, but not GPS readings) to validate this conjecture.

Finally, we find that *Which-place* performs poorly for places a user visited occasionally and stayed for short duration, e.g., *Toxicology* and *McDonald's*. Due to infrequent scanning, Platys didn't find indicative features for such places. Platys confused these places with those the user visited before or after. We will address this limitation in future work via an adaptive scanning approach.

Related Work

Existing place-recognition approaches adopt different formalizations of place.

Spatial approaches rely on spatial features. Ashbrook and Starner (Ashbrook and Starner 2002) and Zhou et al. (Zhou et al. 2007) describe approaches that collect frequent GPS logs (every second and minute, respectively) and apply clustering to extract places. These approaches are expensive. Vu et al. (Vu, Do, and Nahrstedt 2011) apply star clustering on a cooccurrence graph of WiFi access points.

Dynamic approaches consider changes in features. Hightower et al. (Hightower et al. 2005) extract a place by seeking a *stable scan*, which occurs when there is no new cell-tower or WiFi signal seen within a certain period of time. Similarly, SensLoc (Kim et al. 2010) detects a user's entry and departure from a place based on the stability of cell-tower signals.

Staypoint approaches rely on presence within a spatial region for a sufficiently long duration. As discussed in the introduction, their major drawback is the need for tuning radius and duration across places and users. NextPlace

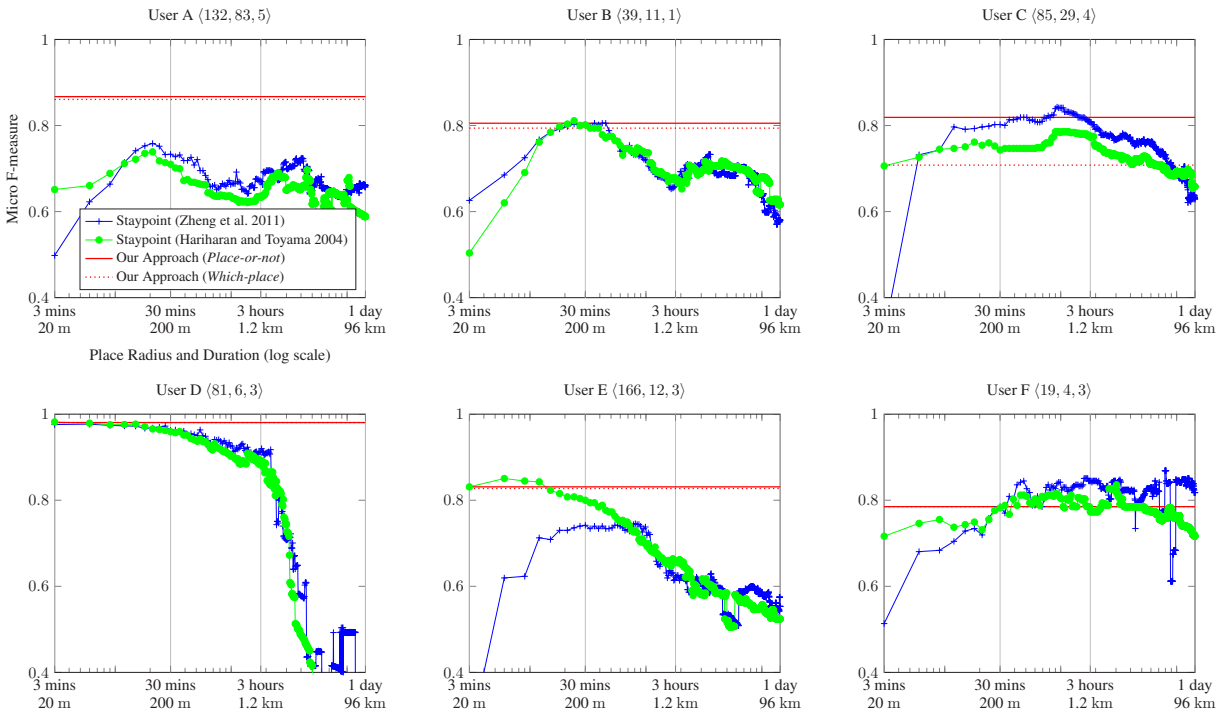

Figure 2: Comparing Platys with two staypoint approaches that distinguish places from nonplace, but do not identify a place. The Platys F-measures are straight lines since it is not tuned to time and distance parameters. We show each user's characteristics as ⟨number of days, number of places labeled, number of US states visited⟩.

Table 1: Precision, recall, and F-measure of some of the places labeled by User A in Figure 3. The table shows the number of timepoints at which User A labeled a place, was actually there at the place (ground truth), and Platys predicted the place.

Place	Labeled	Actual	Predicted	Precision	Recall	F-Measure
Home	69	125	152	0.93	0.91	0.92
Lake Johnson	2	19	31	0.70	0.37	0.48
Lab	55	5	88	0.13	0.40	0.20
Engineering Building	2	9	2	1.00	0.22	0.36
Toxicology	1	5	10	0.20	0.20	0.20
McDonald's	3	4	10	1.00	0.50	0.67
Gym	8	302	324	1.00	0.74	0.85
Friend's place	7	50	48	0.95	0.71	0.81
Nonplace	NA	1,667	1,544	0.86	0.97	0.91

(Scellato et al. 2011) models users' stay at consecutive GPS coordinates as a Gaussian distribution and uses a significance threshold to extract places. Kang et al. (Kang et al. 2005) describe a staypoint approach based on a WiFi data source.

In general, the above techniques work only with particular sensor types. Thus, GPS-based approaches work only outdoors and WiFi-based approaches are not applicable in places such as jogging trails. In contrast, Platys combines multiple data sources in interesting ways and generalizes the applicability.

Conclusions and Directions

Place is an abstract conception of location that is a crucial element of a user's context. A mobile application can employ places to deliver a high-quality user-experience. Platys identifies a user's places despite infrequent and unaligned sensor data. Below, we explore several avenues to enhance the effectiveness of Platys.

The effectiveness of Platys can be improved by incorporating additional data sources. Interesting possibilities exist in incorporating activity sensors (Davies, Siewiorek, and Sukthankar 2008) so as to employ similarity in user activities as a basis for recognizing physically overlapping and even spaceless places. For example, *Home* can be *Office* at

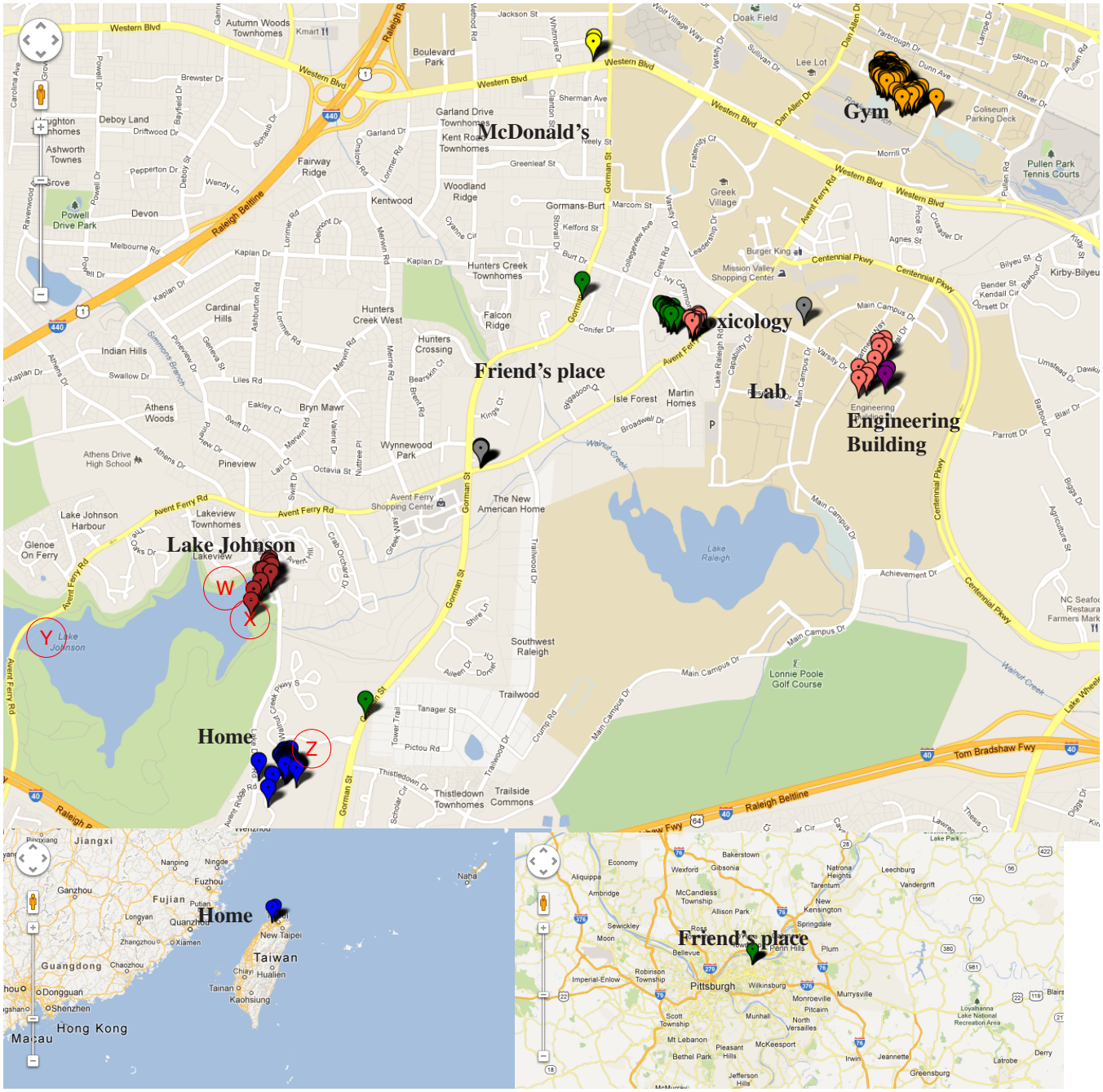

Figure 3: Positions W, X, Y, and Z are included for our running example. The other labels are places identified for User A. Note that the identified places (e.g., *Home* in NC and Taiwan; *Friend's place* in NC and PA) can be geographically disjoint.

times and *Meeting place* may be spaceless. Similar possibilities exist for incorporating social context to distinguish places (Murukannaiah and Singh 2012b). For example, *Office* is associated with *colleagues*, *Home* with *family*, and so on.

Importantly, Platys helps preserve privacy since a user's sensor information and labels are stored locally on the phone. A possible risk arises in revealing the user's position to a web service: this risk may be addressed by generating decoy requests. We defer such considerations to future work.

Although we characterized places as ego-centric, users may share their conception of places with their social circles. In such cases, participatory sensing can help Platys conserve energy and reduce user effort in labeling. For example, if *Lab* is a shared place for all lab-mates, only one of them need label the place and only one device need sense the data. However, such cooperation must preserve users' privacy. Finally, another extension to Platys could be to build a predictive mobility model of a user across places. Such a model, which can predict the next place of a user, can be valuable in place-aware mobile applications.

Acknowledgments

Thanks to the National Science Foundation for support under grant 0910868.

References

Ashbrook, D., and Starner, T. 2002. Learning significant locations and predicting user movement with GPS. In *Proceedings of the 6th International Symposium on Wearable Computers (ISWC)*, 101–108. Seattle, WA, USA: IEEE Computer Society.

Davies, N.; Siewiorek, D. P.; and Sukthankar, R. 2008. Activity-based computing. *IEEE Pervasive Computing* 7(2):20–21.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* 39(1):1–38.

Hariharan, R., and Toyama, K. 2004. Project Lachesis: Parsing and modeling location histories. In *Proceedings of the 3rd International Conference on Geographic Information Science (GIScience)*, 106–124. Springer.

Hightower, J.; Consolvo, S.; LaMarca, A.; Smith, I. E.; and Hughes, J. 2005. Learning and recognizing the places we go. In *Proceedings of the 7th International Conference on Ubiquitous Computing (Ubicomp)*, volume 3660 of *Lecture Notes in Computer Science*, 159–176. Tokyo, Japan: Springer.

Kang, J. H.; Welbourne, W.; Stewart, B.; and Borriello, G. 2005. Extracting places from traces of locations. *SIGMOBILE Mobile Computing Communications Review* 9(3):58–68.

Kim, D. H.; Kim, Y.; Estrin, D.; and Srivastava, M. B. 2010. SensLoc: sensing everyday places and paths using less energy. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 43–56. Zurich, Switzerland: ACM Press.

Lin, D. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning (ICML)*, 296–304. Madison, WI, USA: Morgan Kaufmann.

Murukannaiah, P. K., and Singh, M. P. 2012a. Platys: An empirically evaluated middleware for place-aware application development. http://www.csc.ncsu.edu/faculty/mpsingh/papers/tmp/Platys-developer.pdf.

Murukannaiah, P. K., and Singh, M. P. 2012b. Platys Social: Relating shared places and private social circles. *IEEE Internet Computing* 16(3):53–59.

Scellato, S.; Musolesi, M.; Mascolo, C.; Latora, V.; and Campbell, A. T. 2011. NextPlace: A spatio-temporal prediction framework for pervasive systems. In *Proceedings of the 9th International Conference on Pervasive Computing*, volume 6696 of *Lecture Notes in Computer Science*, 152–169. San Francisco, CA, USA: Springer.

Vu, L.; Do, Q.; and Nahrstedt, K. 2011. Jyotish: Constructive approach for context predictions of people movement from joint Wifi/Bluetooth trace. *Pervasive and Mobile Computing* 7(6):690–704.

Yang, Y., and Liu, X. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 42–49. Berkeley, CA, USA: ACM Press.

Zheng, Y.; Zhang, L.; Ma, Z.; Xie, X.; and Ma, W.-Y. 2011. Recommending friends and locations based on individual location history. *ACM Transactions on the Web (TWEB)* 5(1):5:1–5:29.

Zhou, C.; Frankowski, D.; Ludford, P. J.; Shekhar, S.; and Terveen, L. G. 2007. Discovering personally meaningful places: An interactive clustering approach. *ACM Transactions on Information Systems (TOIS)* 25(3):12:1–12:31.