Improving Wireless Positioning with Look-ahead Map-Matching

Kipp Jones, Ling Liu
School of Computer Science
College of Computing
Georgia Institute of Technology, Atlanta, GA, USA
{kippster, lingliu}@cc.gatech.edu

Farshid Alizadeh-Shabdiz
VP of Technology Development and Research
Skyhook Wireless Inc., Boston, MA, USA
farshid@skyhookwireless.com

Abstract— Determining the location of mobile devices is a necessary system function for a growing number of location based services (LBS). The most popular method for location determination is GPS, however GPS has known accuracy and environmental limitations, many of which are exacerbated in dense urban areas. Wireless Positioning Systems (WPS) on the other hand, demonstrate location accuracy largely inverted to that of GPS – the denser the urban setting, the more accurate the location estimation is in general. Large-scale WPS differs from satellite based positioning in at least two aspects; first, wireless positioning systems typically derive their location estimates based on observed beacon locations such as through wardriving, and second, WPS lacks a mechanism to maintain highly synchronized clocks. This results in lower accuracy and a lower confidence factor in the use of wireless positioning. This paper presents a location estimation method that improves location accuracy for WPS through the use of digital map-matching of wardriving data. We have conducted initial experiments to evaluate our map-matching algorithm along with the enhanced location estimation approach and demonstrate its effectiveness for measuring and improving the accuracy of large-scale wireless positioning systems. We demonstrate extensions to a look-ahead map-matching algorithm to improve the accuracy and motivate further research in the area of large-scale map matching.

Keywords: WiFi; Wireless Networks; Access Points; Location Based Services; Map Matching

I. INTRODUCTION

WiFi Positioning Services (WPS) rely on accurately determining the location of wireless access points (APs) or 'beacons' to estimate the physical location of a WiFi enabled device. The process of determining the location of these beacons is at the heart of any system that will rely on these beacons for subsequent location based services.

This research describes a method to improve the AP location estimation process by introducing topological and geographical constraints in the form of street network models to reduce the amount of error. The process of detecting and correcting errors of location estimation based on map information is referred to as map-matching.

The Global Positioning System (GPS) provides the basis for determining the location of WiFi beacons through a process called wardriving. *Wardriving* refers to the act of searching for WiFi networks by moving vehicle. It involves using a car or truck and a WiFi-equipped computer, such as a laptop or a PDA, to detect the WiFi networks. It is well known that GPS readings are not 100% accurate and can have errors due to a number of factors. Location accuracy is generally expressed as an error within a percentage confidence band. For GPS, the horizontal accuracy is within 22 meters 95% of the time [1]. WPS systems have been characterized as having outdoor positioning with median error of 13-40 meters in one test [7] and 46-63 meters in another [11].

There are numerous areas within wireless positioning systems in which accuracy improvements can be made. This includes such things as improved coverage, sophisticated signal propagation models, the use of directional antennas, the inclusion of signal arrival time, and map-matching. This paper looks at the particular problem of improving the estimation of the location of a vehicle during scanning for access points through a technique known as map-matching. This does not preclude the use of other techniques, which are largely complementary.

We describe a look-ahead technique for improving map-matching accuracy. As distinguished from real-time map-matching algorithms, look-ahead map-matching refers to the ability to use *future* GPS readings to help identify the current point in question. In addition to describing the basic look-ahead map-matching algorithm (LAMM), we develop an optimization to further improve the accuracy of the basic LAMM approach based on curve 'smoothness'. Our look-ahead algorithm is an example of a curve-to-curve fitting algorithm. Look-ahead improves the accuracy of route selection by playing forward the GPS readings beyond the current point in question. By using a locally focused process and incrementally culling out untenable routes, the system aims to bound the execution time while maintaining highly accurate results.

In contrast to most existing MM systems that have been focused on real-time navigation improvements, and thus have been limited to using only past and current information for map matching, in our look-ahead map-matching system, we are not

constrained to real-time which enables us to extend previous approaches with look-ahead capabilities, improving the end result by comparing a set of possible arc matches based on future travel information. One of the distinct features of our approach is that we do not depend on global knowledge, rather using local knowledge on an incremental algorithm, to perform look-ahead map-matching.

The rest of the paper proceeds as follows: we first provide some background and systems details, and then describe the basic LAMM algorithm as well as an additional optimization, followed by experimental evaluation. The paper concludes with a brief review of related work and a conclusion.

II. BACKGROUND

Systems for estimating location based on radio beacons have been a topic of pragmatic and academic research for many years. These systems have taken many forms, from the commercially available outdoor GPS [1] and related navigation systems, to indoor location systems such as the Bat System [8], RADAR [2] and Cricket [9]. There are many applications where location information is essential. Although not all applications require the same degree of accuracy, it is obvious that the availability of more accurate information enables more applications to take advantage of the location information.

This research focuses on improving the accuracy of WiFi location systems, but may have relevance to other systems as well.

A. Wardriving and WiFi Positioning

Examples of large-scale WPS include Intel's Place Lab [7][12] and the Skyhook Wireless[1] WPS. These systems rely on scanning the public airwaves for WiFi access points and building a database of location information related to these access points.

To conduct this study, we acquired the rights to analyze a dataset provided by Skyhook Wireless. A fleet of drivers that systematically drive urban areas to scan for 802.11 WiFi access points collects this data. Skyhook Wireless gathered the current dataset during the time between April 2004 and October 2006. This data corresponds to the systematic scanning in several hundred cities throughout the United States and parts of Canada.

The data is logged using a proprietary scanning software package from Skyhook Wireless in a process often referred to as 'wardriving', a term derived from wardialing which was popularized by the movie "WarGames". The software runs on custom configured mobile devices connected to a standard GPS device via serial or Bluetooth communications. During scanning, no connections are established to the access points. This software utilizes commercial access points to automate the upload of scanned data to a central server. Upon upload, the scanning data is processed to produce the correlation of each access point with its GPS location and signal strength information.

The system measures the signal strength and gathers access point information from the radio signal produced by each AP. For each access point, this includes multiple records that include its name or Service Set Identifier (SSID), the Media Access Control (MAC) address and the timestamp when the AP was scanned. Concurrent with the logging of this data, the geolocation in the form of latitude, longitude, number of satellites, and error is captured using GPS. These GPS readings are the subject of the current research.

Various algorithms and methods have been developed to provide better estimation of the location of wireless access points based on signal readings. Several of these are discussed in [6] and [10] and range from fairly simple triangulation to more complex hierarchical Bayesian sensor models [13].

These efforts are all intended to improve the accuracy of the location estimation and have been shown to be useful in certain circumstances. Map-matching to correct for GPS errors is expected to be complimentary to subsequent radio propagation model improvements.

B. Map-Matching

Map-matching is the process by which the sensor information is integrated with digital maps. The digital maps consist of various geometric objects; the primary object of interest for this research is the network of road segments. This digital data includes the geographic coordinates which can be compared against the GPS coordinates to constrain the GPS readings to points that lie on the road network.

Conventional map matching searches digital maps to fit sensor based location information to a given map. This process is commonly visible in GPS navigation systems that plot a traveler's course on a map to aid in navigation. These map-matching algorithms are also used in transportation planning and analysis, route planning, and automated vehicle control.

III. SYSTEM & METHODOLOGY

Due to space constraints, we focus the description of our system on raw data collection, processing of access point location, and overall map-matching. We refer to the system as Access Point Location based Map-Matching system, APL-MM for short. This section outlines the architecture of the APL-MM system and defines the basic terminology used for defining the goal of the map matching problem, an overview of the system architecture, and the structured definition of the map-matching function and the digital map.

A. *System Architecture*

In the context of our problem, map-matching is the process that correlates the sequence of GPS readings detected and collected by wardriving to the shape of a corresponding digital road database. Raw sensor data is introduced into the system from various scanners. This raw data is fused with the

[1] Commercial service available from http://www.skyhookwireless.com/

incoming WiFi access point data to estimate location. Our current implementation utilizes the GPS records to perform the map-matching, producing adjusted GPS records, which can be used to refine the AP location estimation.

Our map-matching system currently contains over 16 million access point beacons related to over 420 million GPS records. The GPS records are captured in 1-second intervals. This dataset represents some 12 person-years worth of driving data. At 25 km/hour (which corresponds to a random sampling of the data), this corresponds to some 2.6 million kilometers of driving data. This likely represents a large portion of the 6.4 million kilometers of public roads in the United States according to the Bureau of Transportation Statistics [20].

The map we use in this study is the entire United States. The scale of the system and the amount of data being studied presents several challenges in terms of data processing and map matching verification. The scale of the system and the datasets also make a global knowledge solution untenable in an operational mode, even to process a daily addition of some 500,000 GPS readings, approximately 125 hours of scan data at 1 second intervals.

B. *Reference Model*

The GPS sensor data consists of timed geographic points plus a scanner ID, referred to as GPS readings. The scanner ID is simply the MAC address of the device that captured the sensor data. Additional GPS information such as the satellite count and horizontal dilution of precision (HDOP) is not currently used.

The combination of scanner ID and timestamp allow for the determination of a 'session' or so-called sub-trajectories as defined in [16]. These sub-trajectories represent contiguous readings and are created by splitting the dataset based on gaps in subsequent timestamps. Sub-trajectories may also be created based on failure of the algorithm to locate an appropriate extension to the current trajectory, due to either extreme GPS error or incorrect map data. In the APL-MM system, each session is represented by a sequence of contiguous GPS readings gathered during wardriving. The process of session determination is accomplished during execution rather than requiring a data pre-processing phase.

The map-matching function translates the sequences from raw GPS readings onto the nearest road in a digital road database, which represents a road network consisting of nodes and links. Nodes represent intersection points and links represent road segments or arcs.

C. *Digital Map Data*

The data we utilize for the road network is based on the Feature Class 'A' road designation in the Tiger data [21]. A road segment (referred to as an arc in the remainder of the document) consists of *2* to *n* points. These points are connected in a serial fashion.

In general, we are most concerned with the end points or "nodes" of a given arc, however for distance calculations, we must consider a piecewise distance calculation measuring the distance from a given point to each segment within an arc.

The road arcs are generally between 10 meters and 100 meters in length with an approximate mean of 50 meters. According to the U.S. Census Bureau [20], the accuracy of the TIGER/Line data is not as accurate as its six significant digits imply. It is intuitive that more accurate data should ease the map-matching problem by reducing error introduced by the digital map, thus we expect the algorithm to perform at least as good using commercially accurate map data.

IV. ALGORITHMS

To better understand the design objectives and algorithm optimizations employed in the APL-MM system, this section presents variations of map-matching algorithms. We first study a simple point-to-curve based algorithm and describe the potential problems with the simple map-matching algorithm. We then describe our design of the basic LAMM algorithm. We introduce an extension to improve the basic look-ahead algorithm by incorporating a better model for the physical constraints.

A. *Simple Distance Based Matching*

This naive map-matching algorithm translates the raw GPS readings onto the nearest road based on a simple distance calculation. Figure 1 shows an example result of applying such a simple algorithm to sequences of GPS readings collected by wardriving. Clearly, the basic implementation of map-matching, which 'snaps' GPS readings to the nearest road, can generate many errors.

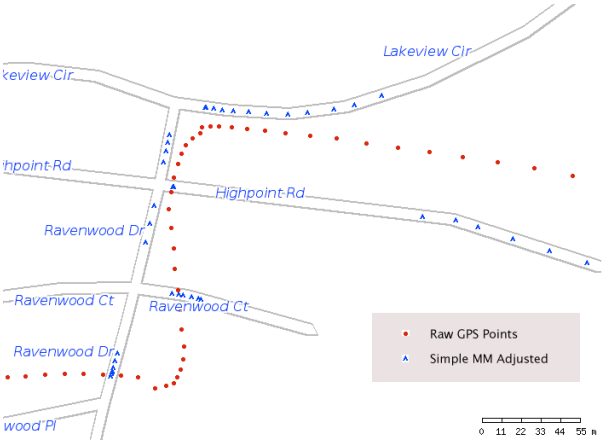

Figure 1. An example of problems with simple map-matching.

Here, the 'tents' represent the adjusted GPS readings based on a simple closest arc calculation. This approach, called Simple Distance Map Matching (SDMM), produced results that were generally good if the only road near the track was in parallel with the wardriver's actual route. However, when a perpendicular road was discovered, the points were snapped to that road rather than to the one on which the driver was traveling.

The figure also illustrates issues when multiple roads run parallel to the track of the traveler, causing the points to be snapped to the incorrect arc. The driver's actual driving route is traveling east on Highpoint Rd. and then proceeding south on Ravenwood Dr. However, the simple distance map-matching produces an incorrect and disconnected path, which is obviously an error caused by map-matching. The type of error associated with the distance from the actual road arc is commonly referred to as the 'cross-track' error.

In addition, this example demonstrates the issue with using a straight linear projection with no interpolation, thus, even when a point is mapped to the correct arc, it is unclear whether the point is in the correct position relative linearly to the arc itself. We refer to this type of error as the 'along-track' error.

By looking closer at the problems encountered by the SDMM approach, we observe that if we can apply global knowledge to look ahead we could continuously correct the map-matching accuracy of the simple MM adjustments to the distance between the raw GPS readings to the nearby road arc. This can help eliminate the cross-track errors. This analysis motivates us to design a map-matching algorithm with look-ahead to reduce these erroneous adjustments.

B. Map-matching with look-ahead

Due to the size of the dataset, a global knowledge algorithm will not be used. However, due to the inaccuracies of simple point-to-curve map matching algorithms and our ability to 'look into the future', a look-ahead greedy algorithm is needed. The proposed approach uses a 'GPS look-ahead' algorithm to build graphs of potential paths that could be traveled. The algorithm uses a tunable look-ahead parameter with a proposed starting limit of 6 points. Details of the algorithm and methods to cull the potential graphs are introduced below and detailed in subsequent sections.

Algorithms which focus on look-ahead suffer from the potential for exponential growth of paths as it creates all possible paths from the current arc out *look-ahead* points forward. GPS points are compared to all arcs in the possible graph in order to cull untenable routes from the graph. This process can create a very large number of possible paths even though a small number of them are actual candidate routes.

One approach to finding additional arcs to add to the candidate routes is by 'walking' the graph of the road network. Alternatively, one can search for road arcs that are 'close' to a given GPS point and choose those that extend any of the existing candidate routes. Our algorithm uses the latter method for extending candidate routes.

For our experiments, we chose a *look-ahead* value of 6, which provides adequate future information while providing minimal impact on performance.

The next step is to find the first set of arcs that match the initial GPS reading. In our algorithm, the 'first' point is actually an average of the first *seed* points to establish a set of starting candidate arcs (where *seed*=5). The candidate arcs

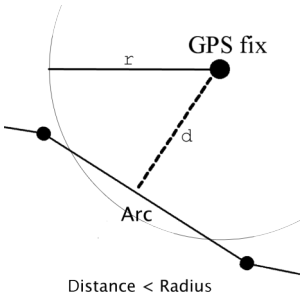

Figure 2. Choosing candidate road arcs.

associated with a GPS reading is defined as those arcs that intersect with the circle range defined by a certain radius r (where r = 40 m) of the GPS reading as shown in Figure 2.

Now we describe the look-ahead map-matching (LAMM) algorithm. Let P_i denote the current location of the GPS sensor sequence, *look-ahead* denotes the look-ahead factor and A_i denotes the i^{th} arc in the set of streets of the road network associated with the GPS reading.

The LAMM algorithm proceeds in seven steps. Step 1 performs the task of obtaining the arc look-ahead using *look-ahead* GPS readings, as described above. Step 2 seeds the algorithm by locating initial candidate arcs based on the average of the first *seed* GPS readings. Step 3 computes the candidate arcs and extends the candidate tracks. Step 4 determines the track curves that best fit the road network by choosing arcs that extend candidate tracks and selecting the best-fit curve set of tracks. The best fit is based on a distance confidence factor for each point up to the look-ahead point and extends work presented in [19]. Step 5 retrieves the next GPS reading. Step 6 is reached at the end of a session (either by running out of GPS records, due to non-contiguous GPS readings, or due to inability to extend the candidate tracks). This step uses the SDMM algorithm to handle the 'tail' of the GPS records since we cannot use the full look-ahead information for these points. Finally, step 7 starts the algorithm over if we have more sessions to process.

A sketch of the LAMM algorithm is given below:

1) **Fill Buffer:** Load the look-ahead buffer with *look-ahead* GPS readings ($P_i - P_{i+\,look\text{-}ahead}$).
2) **Seed Candidate Tracks:** Find starting candidate arc(s) $A_{\text{candidate}}$ where the distance from the average *seed* points is less than $2r$.
3) **Extend Candidate Tracks:** Find next set of arcs by choosing arcs that are connected to previous $A_{\text{candidate}}$ arcs and are less than r from P_i. If P_i can't be matched to any candidate tracks, flag point P_i and skip to step 6 (end of session).
4) **Choose Best Set of Candidate Tracks:** Reduce available paths by:
   a) Choose best-fit curve φ_{Best} from $P_i \rightarrow P_{i+look\text{-}ahead}$.
   b) Eliminate candidate paths that do not have the chosen arc in current tail position.
5) **Get More GPS Readings:** If there are more GPS readings available obtain the next GPS reading and repeat algorithm from step 3.

6) **Find Best Track:** Choose best-fit curve φ_{Best} from all candidate tracks.

7) **Start Next Session:** If there are more GPS readings (i.e. we ended a session but there is more data), begin at step 1.

The algorithm depends on the function φ_{Best} to determine the candidate arcs that best fit the road network. This function is weighted sum of the distance match based on linear distance.

$$\varphi_{\text{Best}} = \max(\varphi_{\text{candidate}}) \text{ where}$$

$$\varphi_{\text{candidate}_i} = \sum_{j=i-lookahead}^{i} \eta \text{ where } \eta = \max\left(\left(1 - \left(\left\lfloor \frac{d}{5} \right\rfloor \bullet .05\right)\right), 0\right) \quad (1)$$

η represents the 'accuracy' of the fit between a given point P_j and its associated arc $A_{candidate}$, i = current GPS reading and d = distance from the point P_j to the arc $A_{candidate}$

This formula assigns a weight between 0 and 1 for each of the *look-ahead* points and the match to the candidate arcs. Points greater than 100m away will not contribute to the summation.

The LAMM algorithm improves on the simple distance map-matching algorithm, though it does not overcome all of the errors. For example, in some instances, the LAMM exacerbates the tendency to snap GPS readings to cross roads that provide a better fit to multiple GPS readings, but clearly is not the actual route that was taken. Figure 3 shows an example of this scenario. The LAMM adjustments on the Eastshore Dr. are computed incorrectly due to the crossing of Highpoint Rd. with Eastshore Dr. The dots located on the road segments represent the corrected readings.

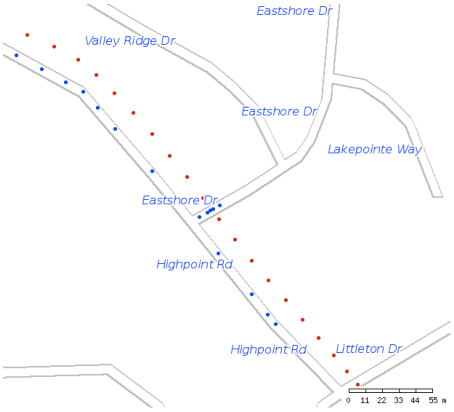

Figure 3. Basic look-ahead map-matching.

V. LOOK-AHEAD MM WITH SMOOTHNESS CONSTRAINT

We have shown that the look-ahead feature does not correct for all errors. This is especially true at intersections and crossroads where the GPS reading may present a smaller distance than from the actual road being traveled. In order to address this problem, in the first prototype of ALP-MM system, we introduce a type of constraint: *smoothness* of MM adjustments. We incorporate this constraint into the Step 4 of the basic LAMM algorithm for determining the best fit φ_{Best} .

A. *Distance-based Smoothness*

We incorporate a new factor, called the *distance-based smoothness*, into our LAMM algorithm. This factor encourages the system to choose arcs in which the distance between subsequent adjusted readings is similar to the distance between their respective GPS readings. This smoothness factor reduces the tendency for 'snapped' readings to jump, thus helping the adjusted readings to conform to a smoother mapping.

An example of the process with smoothness added can be seen in Figure 4 with travel direction indicated by the arrows.

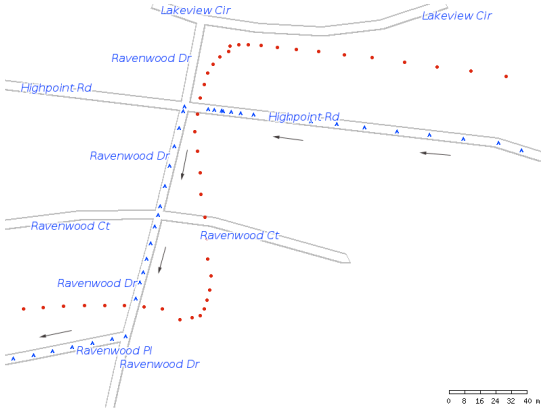

Figure 4. Look-ahead with smoothness constraint.

Smoothness is modeled as a constraint that compares the distance between subsequent GPS readings and the distance between their associated adjusted locations (points). This smoothness constraint encourages adjusted points to have similar offsets from each other as the distance between their corresponding original GPS readings, thus reducing false adjustments. The smoothness constraint is modeled as follows:

for each candidate track τ,

$$\psi_s = \left(\sum_{j\,=\,i\text{-lookahead}}^{i} \eta \text{ where } \eta = \frac{5}{D} \right) \quad (2)$$

η represents the 'smoothness' of the fit between two adjacent points and their equivalent map-matched points such that D = difference between distance from point P_{j-1} to P_j and the distance from the map-matched equivalent P_{j-1}^{MM} to P_j^{MM} and i = current GPS reading.

The smoothness constraint assigns each candidate track with a smoothness score ψ_s based on the relative match between each set of GPS points and their equivalent map-matched points within each candidate track. This equation weights close fitting contiguous points more than those that 'jump' around relative to the original GPS readings.

In the ALP_MM system, we revise the basic LAMM algorithm by applying this constraint to the initial confidence factor that is calculated for each generated arc-point

combination on each candidate route. Concretely, we modify the function φ_{Best} defined in Section IV for determining the road arcs that best fit the road network (the map) by revising the $\varphi_{candidate^i}$ as follows:

$$\varphi'_{candidate^i} = \varphi_{candidate^i} \cdot \psi_S \quad (3)$$

This final $\varphi'_{candidate^i}$ from Eq. (3) is used to determine the best fit for point P_i to some arc A.

VI. Results

Measuring the effectiveness of map-matching requires the knowledge of several pieces of information. First, the actual path followed by the wardriver must be known, and second, the actual set of positions at which the GPS readings were obtained must be known. Without either of these, one cannot *prove* the correctness of the map-matching algorithm. With only the wardriving route, one can ensure that the set of arcs chosen for the final path is correct, but the individual points on the route may not be matched correctly on the digital map. Similarly, with only the known set of sampled positions, an incorrect route may be chosen during the map-matching by, for example, including intersecting or parallel arcs. However, there are a number of techniques that can be employed to measure the quality of the results.

VII. Verification

This section describes the experimental evaluation results of using this map-matching technique on samples of data from the real-world dataset. We examine the impact on the raw data as well as the accuracy improvement in determining the location of an end-user device.

The accuracy of the system can be judged from two perspectives. First, the correction of the GPS readings could be verified using sophisticated assisted GPS combined with dead reckoning. This process is both time-consuming and expensive to perform on a large scale due to equipment costs and the need to re-scan the areas to be verified. A second technique for measuring the improvement is to use the actual WPS system to estimate locations and compare the results with the corrected data. This is the method we promote in this paper. It has the advantage of being able to be automated to scale across the entire database of location information.

Verifying that the map-matching results accurately reflect the actual path and position of the driver during the data acquisition has proven to be quite difficult. One must essentially use a guaranteed correct map-matching algorithm and maintain an independent 'corrected' dataset of the actual path that was taken by the driver. Our validation of the map-matching algorithm is based on knowledge of the route driven by the driver. While the route was known for the data, the actual position en route is not, reducing our ability to correct for along-track error.

We employed a sampling technique to validate the map-matching algorithm. With foreknowledge of the path taken by particular drivers, we manually sampled the output of the map-matching algorithm to verify that the data was adjusted correctly. On average, the algorithm matched between 88% and 92% of the GPS points correctly, while the remaining points were not adjusted. We now describe the impact on end-device location estimation and outline the assumptions that are made.

A. Experimental Technique

Two areas were chosen to test the system. The first included an area in Boston called Boylston that had relatively 'clean' data due to minimal obstructions. The second test area, in Chicago, had less reliable GPS data due to the dense urban setting and a more complex road network. Table 1 presents relevant information about each test data set. The MM Match column represents the percent of GPS points that the map-matching algorithm successfully matched to a road segment.

TABLE 1. TEST REGION CONFIGURATIONS.

Location	Area	Road Segments	GPS Points	MM Match
Boylston	3 km^2	539	91,287	88.3%[2]
Chicago	11 km^2	2752	15,591	91.7%

The validation relied on a data set that was gathered by separately driving the area in question. The data was manually matched to map segments with interpolation of points between segment endpoints. This data then represented '*ground-truth*' for the end-user device. Using this for comparison, we analyzed how the WPS system performed using the original GPS data versus the location estimation based on the map-matched data. In other words, the WPS system itself was used to measure the accuracy of its output.

Figure 5 shows the location estimation for an end-user device based on the original GPS data mapped against the estimation based on the map-match corrected data. Notice that neither sets of data are completely matched to the map – this is due to the error in the WPS system, as the results are after the WPS system has estimated an end user device, not the actual GPS data.

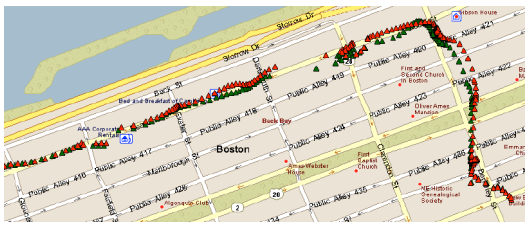

Figure 5. Example of end-device estimation error. In aggregate the map-matched data led to more accurate end-device location estimates.

[2] The lower match ration for Boylston is partially due to a section in which newer roads were not included in the map data set.

Figure 6 illustrates the improvement that was measured based on the map-matching algorithm. The cumulative distribution function (CDF) is use to indicate the probability that a location estimation was within the error bound indicated on the X-axis.

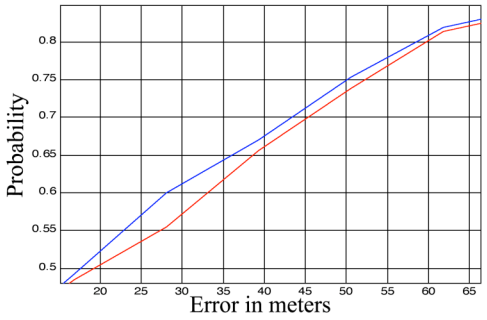

Figure 6. Cumulative distribution function for Boylston. Top line represents the corrected data while the lower line represents the original data.

In this instance, we demonstrate the error distance for both the original GPS data (lower line) and the map-matched data (upper line). Results that skew the function to the left of the graph represent improved accuracy for the data set. For example, 60% will be within 28 meters for corrected data and within 33 meters for the original data. On average the map-matched data (top line) showed an improvement of 2.7 meters, averaging approximately 5% improvement for each GPS point. The 5% improvement is considered a significant improvement for this particular region due to the relative accuracy of the GPS data.

Our second battery of tests involved a data set from downtown Chicago (an area with known GPS errors). In terms of the distance of error, the addition of map-matching decreases the average error distance significantly from nearly 84 meters to just under 73 meters.

Again, we take a look at the cumulative distribution function for Chicago shown in Figure 7.

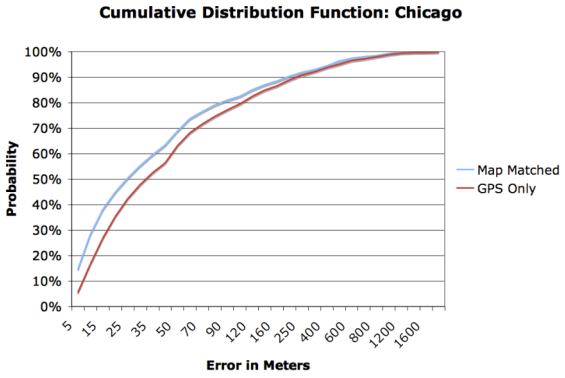

Figure 7. Cumulative distribution function for Chicago.

The two curves shown in Figure 7 depict the original GPS (lower curve) data compared against the adjusted GPS data. The Chicago test indicates potential overall accuracy improvement of 13%.

The improved results were based on adjusting only one of the many driver tracks in each region. Thus, we can expect this to be a lower bound on the improvements – the more tracks that are corrected should, intuitively, continue to improve the location estimation accuracy.

VIII. Related Work

Map-matching algorithms come in several flavors including point-to-point matching [5] in which the GPS reading is simply matched to the closest 'node' or 'shape point' in the map network. This method is sensitive to error in both the GPS positioning as well as the map data.

Point-to-curve matching provides another alternative geometric approach [5]. Rather than matching only to the nearest point as specified in the map network, the navigation point is 'snapped' to the closest curve – generally an arc as defined in the digital map. As we will demonstrate below, this method allows for error in dense urban settings or where the GPS error is large.

More advanced algorithms incorporate additional information. One such class of methods is the curve-to-curve approach in which the trajectory of the navigation information is matched to the trajectory of the map data.

Previous efforts have relied on GPS signals as the primary source, adding dead reckoning and map-matching when necessary. This research examines the use of map-matching to reduce error for all GPS readings (assuming map information is available for the given area). However, as warned by Quddus in [15], "If a good digital network map is not used in the MM process, the positions estimated from the MM process may be worse than the positions from stand-alone GPS". Part of this research includes the verification of improvement based on the MM process.

Ochieng, Quddus and Noland [14] provide a good overview of various map-matching algorithms, while Wenk, Salas, and Pfoser [16] provide more detail relevant to global curve-matching algorithms, which are related to the work presented here. There are several examples of systems that use global or look-ahead knowledge [3][18]. These approaches provide the basis from which our research will extend with a focus on providing an efficient algorithm with high accuracy. [11] provides a good discussion on the problem and issues with accurately locating access points and the resultant effect on localization based on inaccurate access point location information.

The look-ahead capability is possible due to the post-processing nature of this problem. Real-time map matching algorithms have focused on determination of the position based on previous and current information, but were confined by the nature of temporal constraints, i.e. they didn't know the future. While most post-processing algorithms have been devised using global knowledge and full path exploration.

Our look-ahead algorithm can be viewed as a hybrid of Brakatsoulas et al's [4] Incremental algorithm and Wenk et al's

[17] Adaptive Clipping algorithm. The unique feature of our approach is the look-ahead based curve-to-curve fitting algorithm, which improves the reliability estimation with look-ahead, essentially playing forward the GPS readings beyond the current point in question. By using a locally focused algorithm and incrementally culling out untenable routes, the system is able to bound the execution time to a linear function while maintaining highly accurate results.

IX. CONCLUSION

Outdoor wireless positioning systems based on WiFi access points, though still in its infancy stage, has shown a growing promise as a potential complementary technology to GPS and other location services. Understanding the accuracy of such a system and determining methods to improve that accuracy are likely to enhance the usefulness of WiFi positioning systems in providing location based services.

This paper introduces the notion of look-ahead map-matching and presents a location estimation method to improve the quality of wardriving data. Our WiFi location estimation approach consists of a constraint-based look-ahead map-matching algorithm. Our experimental results have shown that the WPS empowered with our map-matching method improves location estimation and can quantify the accuracy of the WiFi location estimation from wardriving data by reducing GPS error via the utilization of a digital map database.

ACKNOWLEDGMENT:

This research is partially supported by Skyhook Wireless, Georgia Tech Tennenbaum Institute, grants from NSF SGER, NSF CyberTrust, NSF CSR, NSF ITR, IBM SUR grant, and IBM Faculty Award. Portions of the described system are covered by patent pending receipt number 15528274.

REFERENCES

[1] Assistant Secretary of Defense for Command, Control, Communications, and Intelligence (2001). Global Positioning System Standard Positioning Service Performance Standard. From http://www.navcen.uscg.gov/gps/geninfo/2001SPSPerformanceStandardFINAL.pdf.

[2] Bahl, P., Padmanabhan, V.N. (2000). RADAR: An In-building RF-based User Location and Tracking System. IEEE Infocom 2000 (March 2000), vol. 2, pp. 775-784.

[3] Battiti, R., Lo Cigno, R., Sabel, M., et al. (2005). Wireless LANs: From WarChalking to open access networks. In *Mobile Networks & Applications* 10 (3): 275-287 JUN 2005.

[4] Brakatsoulas, S., Pfoser, D., Salas, R., and Wenk, C. (2005). On map-matching vehicle tracking data. In *Proceedings of the 31st International Conference on Very Large Data Bases* (Trondheim, Norway, August 30 - September 02, 2005). Very Large Data Bases. VLDB Endowment, 853-864.

[5] Bernstein, D. and Kornhauser, A. (1996). An introduction to map matching for personal navigation assistants. Technical report, New Jersey TIDE Center Technical Report, 1996.

[6] Byers, S., Kormann, D. (2003). 802.11B Access Point Mapping. In *Communications of the ACM*, May 2003, Vol.46, No. 5.

[7] Cheng, Y.-C.; Chawathe, Y.; Lamarca, A.; Krumm, J. (2005). Acuracy Characterization for Metropolitan-scale WiFi Localization. In *Proceedings of the Third International Conference on Mobile Systems, Applications, and Services* (MobiSys 2005), 2005, 233-45.

[8] Harle, R.K., Hopper, A. (2005). Deploying and Evaluating a Location-Aware System. In *Proceedings of the Third International Conference on Mobile Systems, Applications, and Services* (MobiSys 2005), 2005, 219-32.

[9] Hida, K., Mizutani, M., Miyamaru, T., Mineno, H., Miyauchi, N., Mizuno, T. (2006). Design of goods tracking system with mobile detectors. 1st International Symposium on Wireless Pervasive Computing, v 2006, 2006 1st International Symposium on Wireless Pervasive Computing, 2006, p 1-6.

[10] Hightower, J., LaMarca, A., Smith, I.E. (2006). Practical lessons from place lab. In *IEEE Pervasive Computing*, v 5, n 3, July-Sept. 2006, p 32-9.

[11] Kim, M., Fielding, J., Kotz, D. (2006) Risks of using AP locations discovered through war driving. Lecture Notes in Computer Science, v 3968 LNCS, Pervasive Computing - 4th International Conference, PERVASIVE 2006, Proceedings, 2006, p 67-82.

[12] LaMarca, A. et al., (2005). Place Lab: Device Positioning Using Radio Beacons in the Wild. In *Proc. 3rd Int'l Conf. Pervasive Computing* (Pervasive 05), LNCS 3468, Springer, 2005, pp. 116-133.

[13] Letchner, J., Fox, D. and LaMarca, A. (2005). Large-Scale Localization from Wireless Signal Strength. In *Proceedings of the National Conference on Artificial Intelligence* (AAAI 2005).

[14] Li, Z., Chen, W. (2005). A new approach to Map-matching and parameter correcting for vehicle navigation system in the area of shadow of GPS signal. In *Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems*, Vienna, Austria, September 13-16-2005.

[15] Papadimitriou, C. H. (1967). On the complexity of edge traversing. In *Journal of the ACM 23,* 3 (July 1976), 544-554.

[16] Quddus, M.A. (2005). Validation of map matching algorithms using high precision positioning with GPS. In *Journal of Navigation*, v 58, n 2, May 2005, p 257-71.

[17] Wenk, C., Salas, R., Pfoser, D. (2006). Addressing the Need for Map-Matching Speed: Localizing Global Curve-Matching Algorithms, ssdbm, pp. 379-388, 18th International Conference on Scientific and Statistical Database Management (SSDBM'06), 2006.

[18] Wuk Kim; Gyu-In Jee; JangGyu Lee (2000). Efficient use of digital road map in various positioning for ITS. Position Location and Navigation Symposium, IEEE 200013-16 March 2000 Page(s):170 – 176.

[19] Yang, D., Cai, B., Yuan, C. (2003). An improved map-matching algorithm used in vehicle navigation system. *Intelligent Transportation Systems*, 2003. Proceedings. 2003 IEEE Volume 2, 12-15 Oct. 2003 Page(s):1246 - 1250 vol. 2.

[20] U.S. Department of Transportation, Bureau of Transportation Statistics. http://www.bts.gov/publications/state_transportation_profiles/state_transportation_statistics_2005/.

[21] Tiger Data Reference available at : http://www.census.gov/geo/www/tiger/tiger2005se/TGR05SE.pdf