Elastic Pathing: Your Speed is Enough to Track You

Xianyi Gao, Bernhard Firner, Shridatt Sugrim,
Victor Kaiser-Pendergrast, Yulong Yang, Janne Lindqvist
Rutgers University

ABSTRACT
Today, people have the opportunity to opt-in to usage-based automotive insurances for reduced premiums by allowing companies to monitor their driving behavior. Several companies claim to measure only speed data to preserve privacy. With our elastic pathing algorithm, we show that drivers can be tracked by merely collecting their speed data and knowing their home location, which insurance companies do, with an accuracy that constitutes privacy intrusion. To demonstrate the algorithm's real-world applicability, we evaluated its performance with datasets from central New Jersey and Seattle, Washington, representing suburban and urban areas. Our algorithm predicted destinations with error within 250 meters for 14% traces and within 500 meters for 24% traces in the New Jersey dataset (254 traces). For the Seattle dataset (691 traces), we similarly predicted destinations with error within 250 and 500 meters for 13% and 26% of the traces respectively. Our work shows that these insurance schemes enable a substantial breach of privacy.

Author Keywords
location privacy; elastic pathing; usage-based automotive insurance; destination prediction

ACM Classification Keywords
H.4.m Information System Applications: Miscellaneous; K.4.1 Computers and Society: Public Policy Issues—*Privacy*

INTRODUCTION
In the past, automotive insurance companies did not have a large amount of information about their customers. This meant that all customers would pay similar prices, despite potentially large variations in their driving habits. Recently, technological advances have created ways to observe driving behavior. Taking advantage of this technology, some US-based insurance companies [25, 28, 31, 34, 35] now offer consumers insurance policies with the option of installing a monitoring device into their vehicle. The insurance companies state that they use this data to identify safe drivers, who will then be charged lower premiums. The reduced price is used to entice customers to opt in to being monitored. However, unlike a truck driver who is being monitored while doing a job, these consumers will be monitored during all of their daily activities, both public and private.

Although some of these monitoring devices are based upon GPS information and offer no privacy protection, such as On-Star [27], many other devices and insurance programs are advertised as being privacy-preserving. As an alternative to GPS-based tracking, some of these devices log speed information instead. In this paper, we demonstrate that logging this timestamped speed data is also not privacy-preserving, despite the insurance companies' claims.

The privacy problem introduced in this paper is significant because the data collection by insurance companies is an always on activity. Data may be logged forever, and thus, deducing location data from speed traces represents a huge breach of privacy for drivers having these types of insurance policies. Even if insurance companies are not currently obtaining location traces from their data today, this does not guarantee that it will not present problems in the future. The data is not considered to be sensitive data, and therefore is also unlikely to be treated as sensitive. This means that the data may eventually be obtained by any number of antagonists that do know how to process the data traces to obtain location information. For example, it is foreseeable that law enforcement agencies

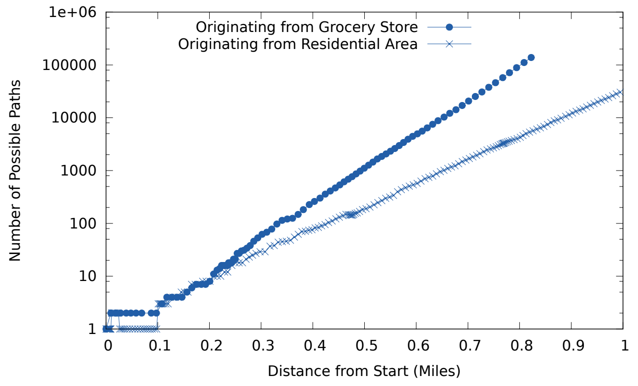

Figure 1. Number of path choices within a given distance of a starting location. The pattern of growth on a log-linear scale shows that the number of paths increases exponentially, making exploration of all possible paths infeasible. Within one mile there are over 10^6 paths diverging from a grocery store and over 10^4 paths from a residential area. Locations central to major transportation routes, such as a grocery store, will see a higher increase in paths compared to a residential location. Residential roads are also more likely to lead to a dead end: in this example the number of paths from the residence actually falls at the beginning because only one path does not dead-end immediately.

would request the information as they have done with other driving related data, for example, electronic toll records [2].

Breaching user privacy with just a starting point and speed information is a difficult task, otherwise insurance companies would not claim that the information being gathered is privacy-preserving. Indeed, matching a single speed trace to all of the roads in a country or state is intuitively extremely difficult. However, the home address of a person is available to insurance companies, making a starting location available for some paths.

It is not obvious that speed data and a starting location are sufficient to reproduce an exact driving path. Speed data does not indicate if a person is turning at an intersection or merely stopping at a stop sign or red light, thus, multiple alternative pathways exist that match some of the speed data. Of course multiple routes can be explored. However, even within just a few minutes drive of a person's home, there may be thousands of turns the person could have taken, so exploring every single possible path is infeasible. The growth of possible paths within a distance of just one mile from a starting location is illustrated by Figure 1. Within one mile, there are over 100,000 possible paths the driver could have taken when the trip starts from a grocery store.

A general approach to reducing the space of possible paths would be to build an error metric for all paths and choose the path that minimizes the error. Due to the rapid growth rate of paths, the number of possible paths at the end of a trip is very large. Simple solutions, such as greedy algorithms, will end in failure without even finding any solution; for instance when the path chosen by a greedy algorithm suddenly comes to a dead end.

This paper makes two major contributions: 1) we present a novel algorithm – *elastic pathing* – to extract location traces from speed data and starting locations, and 2) we implement this algorithm and test it on real world traces to show how the data collected by many insurance companies is not privacy-preserving despite their claims. To the best of our knowledge, we are the first to present such a study with a comprehensive set of real-world traces. We also show that even without predicting path endpoints with 100% accuracy, long-term data collection allows an antagonist to eventually identify physical regions that are frequently visited. The combination of our algorithm with analysis of long-term driving traces can obtain private information from what is currently considered privacy-preserving data. As minor contributions, we collected real-world traces over three months in central New Jersey that serve as one testing dataset for suburban areas, we extracted all driving traces from a Microsoft Research resource [30] to serve as the testing dataset for urban areas, we offer suggestions for further work that may improve path prediction further, we discuss approaches that failed in order to help the community to avoid them, and provide a discussion of privacy-preserving alternatives to the collection of speedometer data that do not require cryptography to implement.

PRIVACY THREATS OF USAGE-BASED AUTOMOTIVE INSURANCE

In several countries, including countries in North America, Europe and Japan, some automotive insurance companies [41] have introduced usage-based insurance programs. In addition to basing insurance premiums on the characteristics of the car and history and knowledge about the drivers, usage-based insurance programs also take into account how and when the car is driven. The incentive to participate for insurance policy holders is a potential discount to their premiums for good driving behavior.

In this paper, we focus on a recent type of a program, which is enabled by collecting speed (and possibly other) data directly from the car. The claim is that by collecting speed data instead of GPS location data (as in other programs [41]), these approaches are more privacy-preserving. There are several insurance companies in the US that use this method. For example, the Snapshot device provided by Progressive and the DriveWise device provided by Allstate will record vehicle speed data while not collecting GPS locations [25, 34]. The data in these programs is collected by connecting a device (provided by the insurance company) to the car diagnostic port OBD-II [36]. Since 1996, OBD-II has been made mandatory for all cars sold in the United States [36].

The data that is collected varies with each insurance company, but in the cases discussed above, it includes time and speedometer data. The devices will regularly upload this data to the insurance company's servers via a mobile data link. The driving habits that some insurance companies state that they are interested in include, for example, "How often you make hard brakes, how many miles you drive each day and how often you drive between midnight and 4 a.m." There is a very important subtlety here. For example, Progressive states that they do not collect the data about when people are speeding. This has caused some members of the public to interpret the statement that speed data is not being collected at all. Instead, they *do* collect speed data, but it is argued that since they do not know your location, they would not know if you are speeding.

To analyze the privacy threats of these programs we need to know the sampling rate of the data collection devices. Unfortunately, the companies do not directly disclose on their web sites how often they sample speedometer data. However, Allstate [25] includes the following information "hard braking events are recorded when your vehicle decelerates more than 8 mph in one second (11.7 ft/s^2) [and] extreme braking events are recorded when your vehicle decelerates more than 10 mph in one second (14.6 ft/s^2)." Because of the requirements for signal reconstruction, we know that the sampling rate must be faster than the duration of the feature the insurance company wishes to observe. In the United States, federal vehicular safety regulations [43] mandate that vehicles traveling at 20 mph must be able to brake to a full stop in 20 feet at a deceleration rate of 21 feet/second^2. Since we know what kind of events the companies are interested in and the maximum performance of vehicles on the road, we can simply estimate the bounds on sampling rates based upon the amount of information required to detect these events.

The events that should be detected occur over a time interval of just one second ($\tau = 1$). Using the Nyquist sampling theorem, we know that they must sample at twice this rate to detect these features, a rate of two samples per second. This is a lower bound on any feature detection, and companies may sample at significantly higher rates to detect other features. Consider drivers that keep only a very short following distance to the cars in front of them. These drivers will often tap their own brakes for very short time durations. If insurance companies wished to detect these events they would need to sample at a rate that was twice the speed of these brief taps. We used much lower sampling rate of one second to ensure the lower bound of sampling rates any company may use.

Summary of Privacy Threats

To summarize, the insurance companies that are using speedometer based measurements are claiming that they cannot find drivers' locations based on their collected data. We will show in further sections that location information is indeed obtainable, contrary to the claim of 100% privacy protection. Further, by finding the correct paths for specific roads, the insurance companies could also deduce if a driver was speeding despite their claims to the contrary.

We will show that our elastic pathing algorithm is able to do more than just identifying the endpoint of a trip – it also eliminates a large number of locations where the trip could not have ended. This is also a serious privacy concern because it allows for the identification of changes in regular routines. For example, if a person usually drives to the grocery store after work but the predicted path goes in a different direction, then it is highly likely that the driver is breaking their routine.

Indeed, previous research has shown that locations (derived from GPS data) can be automatically analyzed to produce profiles of drivers' behavior, social activities and work activities [17]. A comprehensive discussion on the consequences of breaches to location privacy is given by Blumberg and Eckersley [4]. They discuss how location databases can be used by others to ask questions such as "Did you go to an anti-war rally on Tuesday?", "Did you see an AIDS counselor?", "Have you been checking into a motel at lunchtimes?", "Why was your secretary with you?", or "Which church do you attend? Which mosque? Which gay bars?"

DATA

To test the hypothesis that a driving route can be reconstructed from a known starting point and a speed trace we needed to collect a large body of testing data. We required both a ground truth of GPS data and a sample set of speed data. This data was collected with the approach that the insurance companies use, by connecting a device to the Onboard Diagnostics standard OBD-II connector. To log the required data we used two devices: a GPS-enabled smartphone, and a Bluetooth-enabled ODB-II device. We recorded the timestamped speed data and the corresponding GPS positions simultaneously. We also obtained a much higher volume of GPS-only data, from which we reconstructed the speed traces.

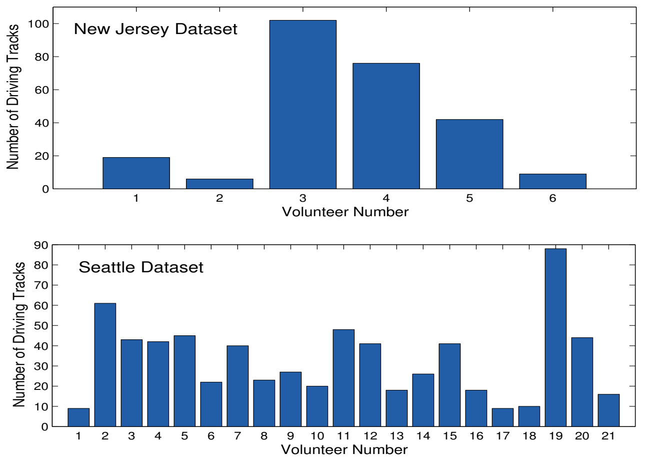

Figure 2. The number of driving traces for each volunteer in the New Jersey dataset (upper figure) and the Seattle dataset (lower figure). The New Jersey dataset has six volunteers with 254 driving traces, and the Seattle dataset has 21 volunteers with 691 traces.

Using the speedometer data from the OBD-II device was straightforward because raw speed data was immediately available. However, some of our data was only available as GPS traces. This data required a small amount of processing to obtain speed values from latitude and longitude pairs. This was a two step process, as outlined below.

First, we used the haversine formula (see appendix) to approximate distances from the raw GPS coordinates. Next, we simply divided this value by the time interval to get an instantaneous speed value for each time interval. Each speed value was then given a timestamp and was converted into the same format as the speedometer traces. We note that we did not see differences in the accuracy of our algorithm between the two approaches of data collection. We also verified that the GPS data approximates actual speedometer data very well, and thus, does not affect our results.

Description of Speed Data

We used two datasets to test our algorithm: the central New Jersey dataset representing suburban areas, and the Seattle city dataset representing urban areas. With nearly one thousand testing traces in total, we believe these datasets represent an appropriately diverse testing set. The idea is that when our algorithm works well on this data, with all of its variety, then it should perform similarly well in many other driving scenarios. In addition to the differences between driving habits of participating drivers, there are also differences in their vehicles. Examples of vehicles that were used in New Jersey dataset collection were small sedans, sports utility vehicles, and a pickup truck. The Seattle dataset is from an external source and we had no control or information of the vehicles used. We feel that this represents a good sample of driving habits and vehicle types. Again, the motivation here is that a pathing algorithm that works properly on all of these traces will work correctly across as a whole. Additionally, the insurance companies know the make and model of the vehicle they are insuring and might be able to use additional information, such as the turning radius of particular vehicles, which we do not use.

Central New Jersey Dataset

We had six volunteers collecting data. Four volunteers collected GPS-only data over a period of three months, and two volunteers gathered both GPS and speedometer data with an OBD-II device over a period of one month. The sampling rate for both the GPS and the speedometer was one second. Figure 2 shows counts of individual driver traces collected. There were 254 data traces that we used for pathing with nearly 1250 miles (nearly 2012 km) in total, with a median trip length of 4.65 miles (7.5 km), minimum trip distance of 0.38 miles (0.62 km), and maximum trip length of 9.96 miles (16.0 km). These traces comprise 46 unique destinations, with more than half of these destinations visited multiple times by individual drivers. 28 locations were visited more than once during data collection and ten locations were visited more than five times. The total driving time for 254 data traces was about 77 hours with an average driving time of 18 minutes for average driving distance of 4.9 miles. The driving areas were mostly not dense urban areas, but residential, suburban and commercial areas connected by highways.

Seattle Dataset

To test our algorithm performance on denser urban areas, we also used an external GPS dataset from Microsoft Research (MSR) [30]. The MSR GPS data was collected by 21 volunteers carrying a GPS logger for about eight weeks in the fall of 2009 in the region of Seattle, Washington. We had looked through several datasets available for access from various research centers and selected this particular dataset for its large size and great potential of extracting valuable driving traces for our testing. However, not all the traces were driving traces, since volunteers also carried around the GPS logger while they were walking or traveling by train or plane. This required the design of a trace filter to pick out all the driving traces. We extracted all the GPS traces that matched the following criteria: the trace consisted of driving in Seattle, and lasted for at least three minutes. This was done in order to have a reasonable distance with all traces and exclude trivial examples.

Other issues with the dataset we needed to consider included the loss of GPS signal while driving. For example, a car driving in a tunnel may lose GPS signal temporally. To ensure the accuracy of the speed calculation using GPS trajectories, two sequential GPS readings should not be separated by more than five seconds. If this happens, they should be considered as two different driving traces. The possibility of losing signal is the only disadvantage of using a GPS device instead of using a speedometer. The dataset from New Jersey did not have this issue.

After trace filtering, 691 traces were extracted with total driving distance of 1778 miles. Figure 2 shows the counts of traces per participant. The mean driving distance per trace was 2.6 miles. The maximum driving distance was 19.9 miles, and the minimum driving distance was 0.59 miles. The average driving time per trace was 11 minutes. Comparing with our own dataset collected by six volunteers, the Seattle dataset has relatively shorter average driving distance. Since it is an urban area, the participants' average driving distance per trip may be shorter than in suburban areas.

ELASTIC PATHING

In this section, we describe the algorithm we use to recreate a person's driving path given a data trace with speed and time pairs, and the origin location. Our approach relies upon two aspects of vehicular travel. First, there are physical limitations to a vehicle's turning radius at high speeds. When a vehicle travels a particular path, it must travel at a speed at or below the maximum speed possible given the radius. Second, people will only stop when they need to, for example, at traffic lights and stop signs, until they reach their final destination. This second assumption is not always true; a person may need to stop because of actions of other vehicles, road construction or accidents. However, we assume that many of the traces are free of these artifacts.

We use data from OpenStreetMap (OSM) to identify possible routes and locations of turns and we find the route that best fits the given speed data. Given the form of the OSM data associated with each road, a path is a sequence of nodes. We can compute the distance between road segments and the angles required to make a turn from the latitude and longitude pairs of the nodes.

A sample in the data trace corresponds to some amount of distance traveled from one node to the next along a path. From the paths of the OSM data, we know what nodes are adjacent, and thus, once a node is reached we know which nodes we can move towards next. For instance, a four-way intersection will have a single node at the intersection and four adjacent nodes. Since we reach the intersection from one direction, we have three possible next nodes from which to choose.

Because of the growth of possible paths shown in Figure 1, all pathing algorithms must have some methods of comparing paths and choosing better paths that more closely match the given speed trace. This also means that the algorithm must keep a list of possible paths, while also either limiting growth of the number of these paths or removing them as they become too plentiful.

Another problem to overcome is that no path perfectly matches the speed data, because drivers swerve around objects in the road, or take turns more widely or sharply than we expect. As the model follows a vehicle's path, the estimation errors accumulate. These errors might cause even the correct path to seem impossible. For example, if the model progresses past the vehicle's actual position along a segment of road, then the vehicle might be going too quickly to make a turn. If we corrected the distance traveled to account for some of these distance estimation errors, then we would find that the speed traces line up perfectly with the turn. Thus, any algorithm must also correct paths as it explores them, trying to take into account these variations in the travel distance.

OpenStreetMap contains road speed minimum and maximum limitations that are useful for path prediction. Since speed data was the only information we were actually processing to predict the destination, having path speed limitations should improve our prediction accuracy by eliminating impossible paths. However, some attempts such as segmenting out steady speed intervals for speed limitation checking, separating highway and non-highway segments, and segmenting transient speed intervals for way entrance speed checking did not work out well.

A simple solution of comparing vehicular speed to the maximum speed limit worked well in the end. If the speed exceeds the maximum speed limit of a road by more than 20 mph, the path is no longer considered possible. After trying different upper bounds, we found the speed limit plus 20 mph provides the best result overall in our testing stage. This seemed to be the upper bound that drivers in our datasets were willing to exceed the speed limit on the roads traveled.

Our Approach: Elastic Pathing

The elastic pathing algorithm is based on compressing or stretching the estimated distance that was traveled as we attempt to match the speed data to the path. After reconciling differences between a section of road and the speed trace we must *pin* the path at that point (which we call a landmark) because any movement would cause a mismatch between the two. For instance, if the driving speed in a trace is reduced to zero, indicating a stop, where there is no intersection we might pull the path forward by some distance to reach an intersection. All points that are pinned cannot be moved since they align with features in the road. We call this approach *elastic pathing* because the stretching and compressing of the speed traces to fit the road is conceptually similar to stretching a piece of elastic along a path while pinning it into place at different points. To help understanding the algorithm, we introduce a small set of definitions.

Calculated Distance The distance a vehicle traveled calculated from the speed and time values in the speed trace.

Predicted Distance The distance along a possible route on the road at a certain time in the speed trace.

Error The difference between calculated distance and predicted distance of a possible route.

Feature A vehicle stop in the speed trace or an intersection in the road.

Landmark A place where the speed trace and road data match, but would become unmatched if any stretching or compressing of the predicted distance traveled occurs. This includes the previously mentioned feature, but also any other place where a mismatch between speed and road data can occur.

We can measure the fit of a path by the amount of stretching or compressing that is needed to be done for the speed data to match the path. The more stretching or compressing along a path, the worse it scores. At each iteration of the algorithm we sort all of the partial paths by their current scores and then explore the path with the best score. That path is advanced until it reaches a feature that requires the pin operation. At this point there may be multiple ways to advance the path so several new paths may be created. Each new path's score is adjusted to reflect the stretching or compressing of the distance traveled from the last pinned landmark. The algorithm proceeds to the next iteration and follows the path with the best score, and thus, the first path to finish cannot be worse than any other path. The pseudocode for this algorithm is given in Algorithm 1.

```
Input: The starting point for pathing, StartNode, and a set of speed
       samples, Samples, and a threshold within the best possible
       score to accept, $\delta$
Result: Complete = list of possible paths within $\delta$ of the best
        possible
1  begin
2      Partial ⟵ {[StartNode ]}
3      Complete ⟵ $\emptyset$
4      while Complete = $\emptyset$ OR
5      Partial.first.error $< \delta \times$ Complete.first.error do
6          P' ⟵ gotoBranch (Partial.pop)
7          join (Complete, {$x \in$ P' | $x$ complete })
8          join (Partial, {$x \in$ P' | $x$ incomplete })
9          sort (Partial)
10     end
11 end
```

Algorithm 1: Pseudocode for the elastic pathing algorithm. Starting with the StartNode, partial paths are added and processed through *gotoBranch* function. In each iteration, partial paths are sorted so that the ones with smaller error have higher priorities during processing.

The most complicated operation done in the *ElasticPathing* algorithm is the *gotoBranch* function, which pseudocode appears in Algorithm 2. The *gotoBranch* function first verifies that the current speed does not exceed the speed limit of the road by more than 20 mph. It then advances the path until it comes to a feature on the map, finds every possible path branching from that feature, and returns those new possible paths. There are two features: an intersection in the road of the path and a zero speed section of the speed samples.

When a path reaches an intersection it explores every possible direction. If the path is going at a speed that can move along the curve in the road then a landmark is set at that location with the *Pin* function. If the curve in the road is too great for the current speed then the distance traveled from the last landmark is compressed with the *compressA* function and stretched with the *stretchA* function. Therefore, when there is a mismatch between speed samples and road data, there are two ways to resolve it, and thus, two new possible paths.

A similar situation occurs when the speed data indicates that the vehicle has come to a stop. If the path is already at an intersection then the landmark is set with the *Pin* function. Otherwise the two solutions are found with the *compressB* and *stretchB* functions. Rather than compressing or stretching the speed trace as in the previous compress and stretch functions, these functions compress or stretch the route data from the last landmark until an intersection is placed at this stop.

After any landmarks are set, the amount of stretching or compressing from the last landmark increases the error of each path. This means that an existing path may have less error than the current paths and should be explored instead. Therefore, all possible branches are returned to the elastic pathing function and the paths are placed in sorted order by their error. The pathing then continues with the new best path.

```
Input: A path, P, speed samples, S, and the current index into the
       speed samples, i
while i < S.length do
    // Match turns at intersections to slower speeds.
    if P at intersection then
        P' ⟵ ∅
        foreach Edge ∈ intersection(P) do
            if S[i].speed ≤ maxSpeed(P, Edge) then
                Pin(P)
                join(P', P ∪ Edge)
            end
            else
                Fore ⟵ compressA(P)
                Back ⟵ stretchA(P)
                join(P', Fore)
                join(P', Back)
            end
        end
        return P'
    end
    // Match 0 speeds to intersections.
    if S[i].speed ≈ 0 then
        while S[i].speed ≈ 0 AND
        i < S.length do
            ++i
        end
        if P at intersection then
            Pin(P)
            return P
        end
        else
            Fore ⟵ compressB(P)
            Back ⟵ stretchB(P)
            return {Fore, Back}
        end
    end
    // Process normally
    if speed higher than way max speed + 20 then
        drop the current path
    end
    else
        update total traveled distance
    end
    ++i
end
```

Algorithm 2: Pseudocode for the *gotoBranch* function used in the elastic pathing algorithm. This code advances a single path until it reaches discrepancy between the speed trace and road. When this happens the path is corrected with compression or stretching and the path is *pinned* at what we call a landmark and the path's error is recomputed. Multiple possible paths may be returned at intersections.

There are several details that we do not show in the pseudocode. For instance, when we check if we are at an intersection, we allow space for the number of lanes in the intersecting road and offset for another car in front of our vehicle. Determining the maximum speed of a turn is done by assuming the maximum allowed incline in the road (8%) [39], the turn radius of the road based upon the number of lanes in the road and typical lane widths [1], and the turn angle equation from Roess et al. [39]:

$$\frac{speed^2}{(15 \times (0.01 \times \text{DEFAULT ELEVATION} + \text{friction}))} \quad (1)$$

In the turn angle equation, friction is the dry coefficient of sliding side friction for tires. Maximum elevation is assumed to be 8% which is the maximum allowed for areas where ice can form on the road. Setting maximum elevation allows the fastest turns, allowing more paths than rejecting them. The turn distance depends upon radius equation given a circular arc or a turn and its chord, which in turn depends upon the number of lanes crossed in the turn. Assume that a driver always makes a smooth curve. We can just find the distance traveled by considering this path as a triangle inscribed in a circle. Flipping the triangle over the x-axis creates a circular segment. The equation for a radius of a circle given a circular segment is $r = h/2 + c^2/(8h)$, where h is the height of the segment (from the chord to the circle's edge) and c is the length of the chord forming the base of the segment. The height of the segment will be the horizontal number of lanes crossed, and the length of the chord is twice the number of vertical lanes crossed.

RESULTS

We ran our algorithm and generated results for both the New Jersey and the Seattle datasets. A Ruby implementation of the elastic pathing algorithm processed all of the New Jersey traces (254 traces) in under 30 minutes on a two-core 2.2 GHz machine, with an average running time of less than ten seconds per trace. For the Seattle dataset (691 traces), it took about one hour with average running time of less than ten second per trace as well. Even without an implementation in a high-performance language, the algorithm is able to process traces far faster than they are generated. Since data analysis could essentially be done in real-time as a vehicle's speed trace is being collected, the limiting factor on the time delay between data recording and path prediction is likely the time overhead of collecting, transmitting, and preprocessing (e.g. matching a trace to a specific driver and their known locations) speed traces. This means that it may be entirely feasible to do near real-time tracking of drivers with just speed data, although the tracking will not be 100% accurate.

New Jersey Dataset Result

For the New Jersey dataset, our algorithm predicted 14% of traces with destination error less than 250 meters and about 24% of traces with destination error less than 500 meters. Table 1 shows the distribution of the number of traces over endpoint error intervals. Figure 3 shows the percentage of individual driving traces with destination error less than 250 meters and 500 meters for each volunteer. The algorithm performance varies on different individual drivers. We refer to the six volunteers as P1 to P6. For P2, our algorithm failed to find any match with endpoint error within 2 miles. In contrast, our algorithm prediction did relatively well for participant P4's driving traces. 20 out of 76 (26%) of traces had endpoint error less than 250 meters. 31 out of 76 (41%) of traces had endpoint error less than 500 meters. These results suggest that some driving styles are easier to localize than others.

Seattle Dataset Result

Our elastic pathing algorithm gave destination prediction error within 250 meters for 13% of traces in the Seattle dataset

Endpoint Error		New Jersey Dataset		Seattle Dataset	
meters	miles	Num. of Traces	Percent	Num. of Traces	Percent
0-250	0-0.16	36	14.17%	90	13.02%
251-500	0.16-0.31	24	9.45%	92	13.31%
501-750	0.31-0.47	14	5.51%	59	8.54%
751-1000	0.47-0.62	11	4.33%	59	8.54%
1001-1250	0.62-0.78	11	4.33%	49	7.09%
1251-1500	0.78-0.93	35	13.78%	44	6.37%
1501-1750	0.93-1.09	2	0.79%	25	3.62%
1751-2000	1.09-1.24	6	2.36%	21	3.04%
2000-2250	1.24-1.40	5	1.97%	29	4.20%
2251-2500	1.40-1.55	6	2.36%	21	3.04%
2501-2750	1.55-1.71	8	3.15%	25	3.62%
2751-3000	1.71-1.86	4	1.57%	17	2.46%
3001-3250	1.86-2.02	6	2.36%	12	1.74%
Greater than 3251	Greater than 2.02	86	33.86%	148	21.42%
Total traces		254	100%	691	100%

Table 1. Results for both central New Jersey dataset and Seattle dataset from our elastic pathing algorithm. Table shows the number of traces having endpoint (or destination) error ranging from 0 to greater than 3.25 km, separated into 14 error intervals as shown in rows. First two columns give endpoint error representations in meters and miles.

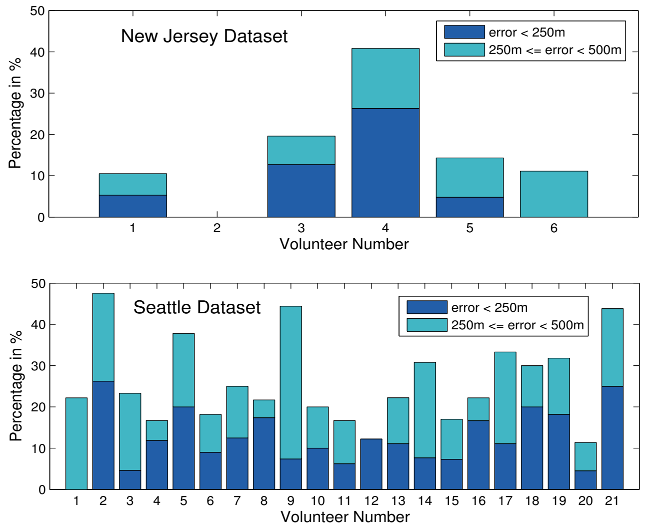

Figure 3. Percentage of individual driving traces that have endpoint error less than 250 meters and 500 meters. For NJ volunteer 2, there is no bar, which means 0% of the traces had the corresponding error.

and within 500 meters for 26% of traces. Table 1 shows the number of traces within each endpoint prediction error interval and the corresponding percentage for the total of 691 traces. For 60% of traces, our algorithm gave the endpoint (or destination) prediction error of less than one mile. However, 22% of traces had endpoint error more than 2 miles. For those traces having large endpoint error, their speed patterns were very noisy. This could be due to heavy traffic which forced cars to stop frequently at positions without any intersection.

Figure 3 shows the percentage of individual driving traces with endpoint error less than 250 meters and 500 meters for each volunteer. Volunteers are referred as V1 to V21 in the following description. For V2, about 26% of the traces had endpoint error less than 250 meters. However, for V1, no trace had prediction error less than 250 meters. V20 had about 5% traces having prediction error less than 250 meters. Besides different driving habits, the percentage difference between V1 and V2 may also due to the difference in number of total traces (see Figure 2). From 21 volunteers, V2's driving traces produced the best results with 26% traces having endpoint error less than 250 meters and 47.5% traces having endpoint error less than 500 meters, while V1 driving traces had the worst result with 0% having endpoint error less than 250 meters and 22% having endpoint error less than 500 meters.

Analysis

The relative accuracy of our approach does not go down with trip distance through our testing on both datasets. Errors were not highly correlated with the path distance. However, in general, shorter trips had smaller variance in error than most of the longer trips. The percent of predicted endpoints within 250 meters of the actual endpoint also does not decrease with distance in our dataset, with trips as long as 10.5 miles still having endpoints correctly predicted to within 250 meters. Our approach clearly does much better than random guessing and almost always correctly identifies at least the general direction of travel. The direction of travel is not a very serious privacy concern though, so we will use another filtering step to screen out noise from incorrectly predicted points.

Our algorithm's performance is dependent on the individual driving habit. Given the same driving location, some individuals may have average speed much higher than others. We assumed that drivers followed the speed limits to some extent to help prune possible paths. This assumption is somewhat incorrect as many drivers speed on highways. However, most drivers obey local (residential) speed limits because of unpredictable driving environments (e.g. children running in the streets). Additionally, local enforcement policies encourage adherence to said speed limits as traffic rules are more strongly enforced in residential areas and fines in these areas are higher. The speed limits in these areas are used as an admissibility criteria for possible paths (e.g. if the driver is traveling at 55 mph (88.5 km/h), it is unlikely that they are on a 25 mph (40.2 km/h) residential road).

We can further take advantage of multiple available traces for each driver to screen out points that may be noise. We note that erroneous predictions are unlikely to repeatedly find the same wrong location and instead usually make mistakes in different locations. Thus, we can use the frequency that locations are predicted as a way to remove poor predictions and focus only on common destinations.

The pathing algorithm is able to do more than just identify the endpoint of a trip – it also eliminates a large number of locations where the trip could not have ended. This is also a serious privacy concern because it allows an antagonist to identify changes in regular routines very easily. For example, if a person usually goes home after work but the predicted path goes in a different direction then it is highly likely the person is breaking their routine.

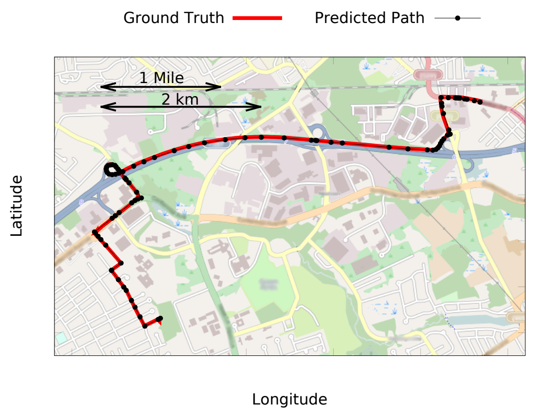

Figure 4. The best possible case, a path from a grocery store to a home with no error. The ground truth is indicated with a solid line while the nodes along the predicted path are shown as dots. In this case the predicted and actual paths match perfectly.

When the algorithm has enough information and a route is very unique, for instance when the spacing between intersections on the road traveled does not match the spacing in any other road, the algorithm can do very well. This is illustrated in Figure 4. However, this case is not normal, and at the very end when the algorithm chooses the correct left turn, it may just as well have chosen to turn right or gone straight. However, even if the algorithm made the wrong choice it would still be within 100 meters of the actual destination.

We believe the probability of error is closely tied to a few factors:

- Homogeneity of roads. If every road seems the same (same speed, similar intersection intervals, etc), especially as in areas built in a grid, then they cannot be distinguished.
- Traces with only slow speeds. Without high speeds to rule out turns and constrain paths to a few major roads, the correct path is indistinguishable from any other.
- Unpredictable stops, caused by traffic, construction, etc.

The largest barrier to improved performance is distinguishing one road from another. If two roads have similar features (e.g. same speed, similarly spaced intersections) then there is little possibilities for the algorithm to distinguish between the two. This is also affected by the traffic pattern of the trip; if a vehicle stops at every intersection because of red lights, then there is a great deal of information about the spacing of those intersections. If, however, the vehicle stops at no lights, then there is no information about the spacing of intersection on the current road. With more prior information we may be able to do better. For example, if we knew the average waiting time at different lights then the waiting time at an intersection might distinguish one road from another even if a vehicle only stops at one or two lights.

This additional information, such as the average wait time at certain lights, or the average traffic speeds of different roads during different times of day, may already be available to insurance companies and this information is not difficult to gather. If the algorithm had this information available, it could score different turning choices with higher accuracy, leading to improved results.

RELATED WORK

Location traces contain a great deal of behavioral information that people consider private [5, 24, 37, 42], which has encouraged a large body of work to concentrate on anonymizing or obfuscating [38, 14, 20, 5, 19] location traces. Krumm [21] has written an overview of computational location privacy techniques, and Zang and Bolot [45] have recently questioned the possibility of releasing privacy-preserving cell phone records while still maintaining research utility in those records. Predicting future mobility patterns and paths from human mobility traces is a well-explored topic, including the following results.

Based on analysis of location data, such as mobile phone cell tower locations, researchers have shown 1) the nature of individual mobility patterns is bounded, and that people visit only a few locations most of the time (e.g. just two [13, 8, 12]); 2) that there is high level of predictability for future and current locations (e.g. [40, 9] – most trivially, “70% of the time the most visited location coincides with the user’s actual location” [40]); and 3) that mobility patterns can also be used to predict social network ties (e.g. [44]). In a more applied domain, GPS traces have been used to learn transportation modes (e.g. walking, in a bus, or driving) [47], for predicting turns, route, and destination [49], for predicting family routines (e.g. picking up or dropping off children to activities) [6], probabilistic models describing when people are home or away [23], and recommending friends and locations [48]. However, none of the above work can be used to discover locations based solely on speedometer measurements and a driver’s starting location.

Related work in the field of dead reckoning does suggest that speedometer traces should have some level of information that can be extracted. Dead reckoning works by using speed or movement data starting from a known location to deduce a movement path. Dead reckoning has typically been used for map building with mobile robots [11] or as an addition to Global Navigation Satellite Systems (GNSS) such as GPS [18, 46]. When supplementing GNSS data, odometer data for dead reckoning might only be used when GNSS information is unavailable, such as when a vehicle passes through a tunnel or an area with many tall buildings. However, this kind of dead reckoning cannot work without frequent location ground truths as there is no perfect method to match speed data to turns in a road.

Map building with robots is interesting because there are often no exact ground truths, only location estimates. Golfarelli et al. [11] describe a method that can assemble maps from arbitrary distance estimates to and from landmarks. The problem of deducing a traveled path from only speed data and a starting point is, in some ways, the reverse of this map building problem: we already have the map, but have difficulty identifying landmarks (in this case turns) from just speed data.

In the specific area of usage-based insurance, the privacy concerns of these insurance programs have been studied before by Troncoso et al. [41]. However, this work dates back to schemes which would send raw location (GPS) coordinates to either insurance providers or brokers, and Troncoso et al. proposed a cryptographic scheme PriPAYD [41] to address the problem. Our work shows that speedometer-based solutions, which were not considered by Troncoso et al., are not privacy-preserving either.

Technically similar to our work are side-channel attacks using accelerometer data from smartphones [3] to infer user location. Projects such as ACComplice [16] and AutoWitness [15] have used accelerometers and gyroscopes for localization of drivers. However, the information from the smartphone can be used to actually detect when turns occur. In contrast, we have only a time series of speed data available. Although this speed data might indicate when a vehicle stops or slows down at an intersection, unlike with gyroscopic data, it does not indicate if any turn is taken.

Finally, the most closely related work to our own is a project to infer trip destinations from driving habits data by Dewri et al. [7]. Their work used time, speed, and driving distance data to estimate destinations of only 30 trips within a single geographic area with error within 0.5 miles, showing how location privacy can be breached with speed data. The Dewri et al. algorithm was essentially a depth-first-search (DFS) that pruned explored paths if they were too long or did not match up to assumed intersections where vehicle speed dropped to zero for several seconds. This approach is an interesting proof of concept, but in areas with a large number of paths a DFS algorithm has to explore a very large number of paths. The authors also needed to include several "slack" heuristics to account for the accumulation of distance error between measured travel distance and actual travel distance, and in a dense urban area would lead to many paths matching a recorded trace. Our work differs by providing a substantially more efficient algorithm that works for a comprehensive amount of traces for both suburban and urban areas. Our dataset consisted of realistic traces collected during people's daily lives, including cases having abnormal speed patterns (such as trips with heavy traffic), and performed path reconstruction over long distances in areas with high densities of roads. We also demonstrated that our approach works across a variety of drivers and in multiple geographical locations, with nearly one thousand traces in total, without relying upon heuristic values for slack and instead incorporating the idea of an error metric that can be compared across traces.

DISCUSSION

Our algorithm provides reliable prediction results for traces having good speed patterns, meaning traces without heavy traffic, and without much random stopping or maneuvering. In addition, driving habits are often regular and trips are usually repeated. Therefore, a set of possible destinations can be built and reduced to a handful since the random artifacts that make a speed trace fail have a low probability of repeating. Given that we know the history of speed data and the driver's home location, we believe our algorithm can be further improved by automatically learning the individual driving behavior and driving environment to have the best prediction for different individuals.

Therefore, the privacy threat is real. Although it is possible for companies to adjust their sampling rates to reduce the potential for privacy breaches, there is no guarantee that someone could not improve upon our algorithm and obtain results better than ours even with a lower sampling rate. Perhaps the best way to improve the privacy protection of these systems is to process data in the car before sending it to a remote server, so that only the end results are ever transmitted and stored.

Failed Approaches

Although path reconstruction may seem simple in concept, it is actually very difficult. The first major difficulty is detecting turns with only speed data. When drivers break and then accelerate, did they make a turn or did they slow down because the person in front of them was turning? If drivers come to a complete stop and then accelerate, did they go straight or turn? When drivers stop at an intersection are they closest to the stop light, or are they behind ten other cars? The speed data alone does not give enough information, and a single incorrectly identified turn will result in an error possibly as large as the entire trip distance. Further, even when a turn is identified from the speed data, if the predicted position of the car is off by a few meters, then a wrong intersecting road may be chosen as the destination of a turn.

In principle, turns could be predicted probabilistically based upon past driver behavior [10]. However, we assume that we do not already know information about an individual driver, so we cannot build a model of the driver's specific driving. We could consider using a model developed using many other drivers in the same area [22], but that has the disadvantage of predicting the most common turns and paths rather than the unique path of an individual. Estimating that people probably shop at a supermarket near them or that they probably work at the largest employer in their town is not necessarily a considerable breach of their privacy. Instead, we identify specific locations that a particular individual visits, and are especially interested in those places that are unique to the individual and not common to many people.

Since the heart of this pathing problem is the lack of turn information, one approach is to treat this as a classification problem. Here, we examine the speed traces to extract features: stop, right turn, left turn, or straight road. If classification would work well, we can assign probabilities to different choices at each intersection and eventually find the highest probability path. However, road conditions and driving habits are very diverse. We found that with our datasets we were unable to classify turn features with high enough accuracy to actually build correct paths. The sheer volume of paths to explore meant that unless the correct path was extremely different from all of the incorrect paths, the algorithm would find some incorrect paths that looked just as good, or better, than the correct path.

Another approach we tried was to switch from a probabilistic model to a model where we identified time windows where turns could occur and did not follow any paths that would make turns outside of these windows. We expected this to reduce the number of paths we needed to explore, perhaps leaving just a few paths at completion. However, because of the accumulation of distance estimation errors, turns in the speed data do not align with the map data so we needed to widen the time intervals where turns were allowed. When the time windows where we accepted turns increased, the number of paths also increased, until we were again left with so many paths that it was unclear which ones best matched the data.

We also tried a dynamic programming approach, keeping track of the score of the best possible path at each node of the ways in the OSM data. However, there is a subtle flaw in this approach. Whatever error metric is used for each path, that error must slowly increase as the path grows in size due to the slight discrepancies in estimated and real travel distance. If there are two possible paths to reach a node, then this dynamic programming approach will tend to favor the shorter path that reaches that node. However, if we explored the path further, we might find that the longer path with a worse score at the previous node better matches the road going forward – but with this dynamic programming approach we would have already dropped that path because it had the worse score. This error would be unrecoverable.

The approach that works must keep track of the best path in terms of time; if two paths have the same error score but one path has traveled through more of the speed samples, then that path is better than the other (up to that point) because it is accumulating less error per unit time.

Privacy-Preserving Alternatives to Speedometer Readings

There are alternatives to collecting speedometer readings which should have better privacy protection. In fact, some usage-based insurance programs are already offering more privacy-preserving products. Companies including GMAC insurance, MileMeter and Corona Direct offer programs that only require the mileage information from vehicles [28, 32, 26]. PAYD coverage provided by Hollard Insurance Company collects mileage information and driver style information [29]. Companies like PayGo Systems collect driving zones instead of fine-grained location, in addition to basic information including mileage and usage time [33].

The trend of companies collecting more and more data about their customers will only increase as sensors and wireless communication become more inexpensive and widespread. The research community is well aware of the privacy risks of such large datasets, but this caution is not widespread in the industrial community. It makes sense that companies promising privacy should open their data collection methods and assumptions in the same way that standardization bodies solicit public comments and new cryptographic standards open themselves up to public testing and scrutiny. Analysis by independent experts would allow companies to verify privacy claims before making assertions.

In the case of automotive insurance companies, data collection should not be a secret practice – the insurance business relies upon building a model of your behavior based on interesting events. A speed trace is not vital to this system, as the real goal is the detection of unsafe driving habits, which can be detected with a more privacy-preserving set of driving features.

CONCLUSIONS

We have demonstrated an effective attack against user location privacy that uses speed data from driving traces and an initial starting location. We have presented the design and implementation of a novel elastic pathing algorithm and used a large corpus of real-world traces to show its accuracy. Our trials showed that the algorithm is very fast to compute, and could also be used for real-time tracking of people just based on their speed data.

Our results show that an entity can gain knowledge about private information about drivers. With two datasets together, there are about one thousand traces. For 26% of all traces, we predicted the correct end point of a trip to within 500 meters. This means that a location visited daily can be identified with a week or more of data and locations visited on a weekly basis could be identified with about a month or more of data.

In time, we expect that improvements will be made to our initial approach. Our work is not directed against any company or organization. However, prior experience has shown that even well-intentioned uses of data can result in losses of privacy, and thus, we wish to highlight potential dangers of this type of data collection. We do not claim that insurance companies are violating policy holder privacy in this way. However, the general principle of any privacy-preserving data collection is to collect only the data that is necessary for a particular application, and no more.

Further information of this project is available at `http://elasticpathing.org/`. The site contains the implementation of our elastic pathing algorithm presented in this paper. The code includes tools to get the needed map data from OpenStreetMap. The site also includes a subset of the traces we collected, which have been explicitly authorized to be made publicly available, and scripts for getting the driving traces from the MSR dataset.

Appendix: Haversine formula

$$\begin{aligned} A &= (sin^2(\frac{\Delta lat}{2}) + sin^2(\frac{\Delta lon}{2})) \times cos(lat_1) \times cos(lat_2) \\ C &= 2 \times atan2(\sqrt{A}, \sqrt{1-A}) \\ Distance &= R \times C \times 0.62 \end{aligned} \tag{2}$$

Where $\Delta lat, \Delta lon$ are the changes in latitude and longitude respectively, and R is the radius of the earth (approximately: 6371 km). The factor of 0.62 is for km to miles conversion.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant Numbers 1228777 and 1211079. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

1. AASHO. *A Policy on Geometric Design of Highways and Streets*, 6th ed. AASHO, 2011.
2. Apuzzo, M. Police increasingly use electronic toll records to solve crime. *The Associated Press* (Dec. 2003).
3. Aviv, A. J., Sapp, B., Blaze, M., and Smith, J. M. Practicality of accelerometer side channels on smartphones. In *Proc. of ACSAC'12* (2012).
4. Blumberg, A. J., and Eckersley, P. On locational privacy, and how to avoid losing it forever, Aug. 2009. `https://www.eff.org/wp/locational-privacy`.
5. Brush, A. B., Krumm, J., and Scott, J. Exploring end user preferences for location obfuscation, location-based services, and the value of location. In *Proc. of Ubicomp '10* (2010).
6. Davidoff, S., Ziebart, B. D., Zimmerman, J., and Dey, A. K. Learning patterns of pick-ups and drop-offs to support busy family coordination. In *Proc. of CHI '11* (2011).
7. Dewri, R., Annadata, P., Eltarjaman, W., and Thurimella, R. Inferring trip destinations from driving habits data. In *Proc. of WPES'13* (2013).
8. Eagle, N., and Pentland, A. S. Eigenbehaviors: identifying structure in routine. *Behavioral Ecology and Sociobiology 63*, 7 (2009), 1057–1066.
9. Farrahi, K., and Gatica-Perez, D. Discovering routines from large-scale human locations using probabilistic topic models. *ACM Trans. Intell. Syst. Technol. 2* (January 2011), 3:1–3:27.
10. Froehlich, J., and Krumm, J. Route prediction from trip observations. *SAE SP 2193* (2008), 53.
11. Golfarelli, M., Maio, D., and Rizzi, S. Elastic correction of dead-reckoning errors in map building. In *Proc. of IEEE/RSJ International Conference '98*, vol. 2 (1998).
12. Golle, P., and Partridge, K. On the anonymity of home/work location pairs. In *Proc. of Pervasive '09* (2009).
13. Gonzle, M. C., Hidalgo, C. A., and Barabasi, A.-L. Understanding individual human mobility patterns. *Nature 453* (June 2008), 779–782.
14. Gruteser, M., and Grunwald, D. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of MobiSys '03* (2003).
15. Guha, S., Plarre, K., Lissner, D., Mitra, S., Krishna, B., Dutta, P., and Kumar, S. Autowitness: locating and tracking stolen property while tolerating GPS and radio outages. In *Proc. of SenSys '10* (2010).
16. Han, J., Owusu, E., Nguyen, L. T., Perrig, A., and Zhang, J. Accomplice: Location inference using accelerometers on smartphones. In *Proc. of COMSNETS'12* (2012).
17. Iqbal, M. U., and Lim, S. Privacy implications of automated GPS tracking and profiling. In *Proc. of The 2nd Workshop on the Social Implications of National Security* (Oct. 2007).
18. Krakiwsky, E., Harris, C., and Wong, R. A kalman filter for integrating dead reckoning, map matching and gps positioning. In *Proc. of IEEE PLANS '88.* (1988).
19. Krumm, J. Inference attacks on location tracks. In *Proc. of Pervasive'07* (2007).
20. Krumm, J. Realistic driving trips for location privacy. In *Proc. of Pervasive '09* (2009).
21. Krumm, J. A survey of computational location privacy. *Personal Ubiquitous Comput. 13*, 6 (Aug. 2009), 391–399.
22. Krumm, J. Where will they turn: Predicting turn proportions at intersections. *Personal and Ubiquitous Computing 14*, 7 (2010), 591–599.
23. Krumm, J., and Brush, A. J. B. Learning time-based presence probabilities. In *Proc. of Pervasive'11* (2011).
24. Ludford, P. J., Priedhorsky, R., Reily, K., and Terveen, L. Capturing, sharing, and using local place information. In *Proc. of CHI '07* (2007).
25. Allstate Insurance Company. drivewise. `http://www.allstate.com/drive-wise.aspx`.
26. Corona Direct. Insurance in kilometre. `http://www.milemeter.com/milemeter2`.
27. General Motors. Onstar. `https://www.onstar.com/`.
28. GMAC Insurance. Gmac insurance low-mileage discount. `http://www.gmacinsurance.com/auto-insurance/smart-discounts/low-mileage-discount.asp`.
29. Hollard Insurance Company. Pay as you drive. `http://www.payasyoudrive.co.za`.
30. Krumm, J. and Brush. A.J. (2009). MSR GPS Privacy Dataset 2009. `http://research.microsoft.com/~jckrumm/GPSData2009`.
31. Liberty Mutual Agency Corporation. Onboard advisor. `http://www.onboardadvisor.com`.
32. MileMeter Insurance Company. Milemeter 2. `http://www.milemeter.com/milemeter2`.
33. PayGo Systems. Paygo. `http://paygo-system.com`.
34. Progressive Casualty Insurance Company. Snapshot. `http://www.progressive.com/auto/snapshot.aspx`.
35. State Farm Mutual Automobile Insurance Company. Drive safe & save. `http://www.statefarm.com/insurance/auto_insurance/drive-safe-save/drive-safe-save.asp`.
36. United States Environmental Protection Agency. On-Board Diagnostics (OBD). `http://www.epa.gov/obd/basic.htm`.

37. Patil, S., and Lai, J. Who gets to know what when: configuring privacy permissions in an awareness application. In *Proc. of CHI'05* (2005).

38. Popa, R. A., Blumberg, A. J., Balakrishnan, H., and Li, F. H. Privacy and accountability for location-based aggregate statistics. In *Proc. of CCS '11* (2011).

39. Roess, R. P., Prassas, E. S., and McShane, W. R. *Traffic Engineering*. Pearson, 2011.

40. Song, C., Qu, Z., Blumm, N., and Barabasi, A.-L. Limits of predictability in human mobility. *Science 327*, 5968 (Feb. 2010), 1018–1021.

41. Troncoso, C., Danezis, G., Kosta, E., Balasch, J., and Preneel, B. Pripayd: Privacy-friendly pay-as-you-drive insurance. *IEEE Trans. Dependable Secur. Comput. 8*, 5 (Sept. 2011), 742–755.

42. Tsai, J. Y., Kelley, P., Drielsma, P., Cranor, L. F., Hong, J., and Sadeh, N. Who's viewed you?: the impact of feedback in a mobile location-sharing application. In *Proc. of CHI'09* (2009).

43. U.S. Department of Transportation. Federal motor carrier safety administration. `http://www.fmcsa.dot.gov/rules-regulations/administration/fmcsr/fmcsrruletext.aspx?reg=393.52`.

44. Wang, D., Pedreschi, D., Song, C., Giannotti, F., and Barabasi, A.-L. Human mobility, social ties, and link prediction. In *Proc. of KDD '11* (2011).

45. Zang, H., and Bolot, J. Anonymization of location data does not work: a large-scale measurement study. In *Proc. of MobiCom '11* (2011).

46. Zhao, L., Ochieng, W. Y., Quddus, M. A., and Noland, R. B. An extended kalman filter algorithm for integrating GPS and low cost dead reckoning system data for vehicle performance and emissions monitoring. *The Journal of Navigation 56*, 02 (2003).

47. Zheng, Y., Chen, Y., Li, Q., Xie, X., and Ma, W.-Y. Understanding transportation modes based on GPS data for web applications. *ACM Trans. Web 4* (January 2010), 1:1–1:36.

48. Zheng, Y., Zhang, L., Ma, Z., Xie, X., and Ma, W.-Y. Recommending friends and locations based on individual location history. *ACM Trans. Web 5* (February 2011), 5:1–5:44.

49. Ziebart, B. D., Maas, A. L., Dey, A. K., and Bagnell, J. A. Navigate like a cabbie: probabilistic reasoning from observed context-aware behavior. In *Proc. of UbiComp '08* (2008).