CONSTRUCTING ROUTE CHOICE MAPS FROM GPS TRACES

Ana Alves, ana@dei.uc.pt Universidade de Coimbra, Portugal

João Rodrigues, jarod@student.dei.uc.pt Universidade de Coimbra, Portugal

Pedro Correia, pamc@dei.uc.pt Universidade de Coimbra, Portugal

Penousal Machado, machado@dei.uc.pt Universidade de Coimbra, Portugal

Francisco C. Pereira, camara@smart.mit.edu Singapore-MIT Alliance for Research and Technology, Singapore

ABSTRACT

Our route choices depend on our own mental map of the city. This assumption, brought from the areas of spatial cognition and cognitive mapping, establishes a distorted graph as the cognitive model that represents our trips. The task of trajectory prediction consists of determining, when a driver is making a trip, which path will be chosen. Our approach consists of transforming the map according to perceived preferences of the driver and then applying a classical route planning algorithm to determine the route. In this paper, we study the process of map distortion according to different parameters and in different contexts. We use GPS logs obtained from volunteers to analyze these effects in the accuracy of trajectory prediction in a real world context.

Keywords: Spatial cognition, cognitive mapping, trajectory prediction, route planning

INTRODUCTION

In our daily lives, much of our driving is routine and it is easy to detect patterns since we frequently go to the same destinations and follow the same routes over and over again. In fact, Krumm and Froehlich (2008) found out that, for drivers observed during at least 40 days, about 60% of their future trips are repeated in the observations. Even if there are shorter or faster ways to reach the destinations, the drivers tend to stick with the same routes used in the past. While this repetition is observed in our habits, it is common to find considerably different sets of trajectory (or route) choices among different people, for any pair of origin and

destination points that are not too close to each other. In fact, this phenomenon has been thoroughly studied, and gender, age, visual-spatial capabilities and level of familiarity with the area have found to be explanatory components. It turns out that each one of us has his/her own cognitive map of the world, a distorted representation of the real map. It results from the compromise between the need of an accurate representation of the world and our memory and reasoning limitations. Such distortion actually changes with time, according to our needs. As we get more acquainted with an area, the distortion becomes gradually corrected; as we forget an area or still barely know it, the distortion is highly affected by the little we know or remember.

Since, in general, the chosen trajectory is the one that optimizes a cost function (e.g. shortest, quickest, with less expected traffic, traffic lights, etc.), we also conclude that the differences between people are not so much on how this search for the best route is made, but on their perception of the traffic network and of its conditions at the moment of choice. Thus, by collecting and analyzing GPS traces of a driver, it should be possible to detect the patterns described above and predict what will be the driver's next destination and route taken in a specific time of a day. This information can then be used to improve the current navigations systems. These devices know how to compute and show the driver the shortest path for a given destination and some are also capable of displaying information about traffic and points of interest (POIs) nearby. However, the current shortest path is not always the preferred by the drivers so, without knowing their behaviours and preferences, the help these devices can provide becomes essentially limited to guide the driver in unknown areas.

During the recent years, the driver's trajectory and destination prediction research fields have been improving rapidly, in part due to the advantages we can take from them in the new car navigation systems. In this paper, we focus on the problem of generating an accurate estimation of what we call the distorted graph of the driver given his/her GPS traces. We propose two similar models that predict the trajectory (or sequence of links) that a driver will choose given an origin, destination and a set of past traces. Our hypothesis is that the accuracy of this prediction is dependent on how the drivers' space perceptions reflect their actual route choices. The validation consists of making the predictions for eleven volunteer drivers from three different cities.

Although it is very tempting to do otherwise, it is not our ambition to reach the real cognitive map, as this would hardly be verifiable, particularly without contextual information (e.g. traffic conditions, traffic light spots). Instead, the distorted graph is a representation of the road network where weights are transformed in such a way that a classical shortest path algorithm (Dijkstra, 1959) eventually yields the actual choices. In other words, by applying Dijkstra algorithm to a pair of known origin and destination, our algorithm should be able to predict accurately the trajectory that the driver would traverse. The distorted graph should thus be a proxy for the cognitive map for the single purposes of trajectory modeling and prediction. We will make no claims in this paper on the correspondence to a specific cognitive map of the driver although we believe that it can be a good indicator in that direction.

A side note should also be made in that the concept of cognitive map is radically different from that of personal map. The former corresponds to the mental image of the map of a city

and its perceived distances and therefore it is very dependent on spatial and visual aspects while the latter is associated to a set of preferred or familiar places and is essentially symbolic. The former works on the links of the graph while the latter is essentially focused on its nodes (and landmarks or significant places).

RELATED WORK

The quest for efficient algorithms that predict a full route based on previous driving behaviour has motivated a number of relevant research works. We will now give a brief overview of current work in this topic in the areas of spatial cognition, route selection and trajectory prediction.

Spatial Cognition

The topic of spatial cognition (sometimes called cognitive mapping) deals with the study of how humans (and other animals) perceive and represent space, particularly very large spatial entities such as towns, cities, neighbourhoods, landscapes, metropolitan areas, environments and the like. Because of their size, such entities can never be seen in their entirety, and consequently one constructs their internal representation by means of visual, as well as non-visual, modes of sensation and information: text; auditory, haptic and olfactory means for example, or by inference (Portugali, 1996). It is almost impossible for a person to perceive and represent such entities in perfection, thus the presence of distortions in this kind of mental representations is unquestionable. *Cognitive Map* is the usual metaphor for the human mental image of the maps of such environments that differ from the real ones because they are distorted in different ways regarding each person.

The implications of such conceptualization of space in terms of trajectory choices have also been thoroughly studied. For example, in a study with 64 volunteers who were asked to choose between two equal-length routes, one going generally north and one south, the results show reliable decision preferences towards the southern option (Brunyé et al., 2010). The authors make further experiments to conclude that north choices are perceptually interpreted as also being “uphill”. In other studies, it is verified that individuals prefer least-angle strategies and with an initial straight segment (Bailenson et al., 2000). Another interesting finding (Lloyd & Heivly, 1987) is that the cognitive map distortion of an individual seems to be correlated with home location. For example, city center locals tend to have a better notion of location of landmarks of the city and a worse notion of absolute distance than inhabitants from the vicinities. In another study performed in some North America cities, Stevens and Coupe (1978) realized that people tend to store in their memories not the exact location of all cities, but rather the relative location of the states, calling this type of distortion hierarchical organization. In 1981, Tversky also detected other types of distortions like Rotation (people tend to rotate spatial locations so that they will correspond to the exact N-S frame of reference) and Alignment (two nearly-aligned locations tend to be remembered as more closely aligned than they actually were). In fact, judgements of spatial relations might be affected by political (Maki, 1981) and semantic (Hirtle & Mascolo, 1986) reasons too.

An important generic finding is that people tend to choose routes that contain the fewest number of landmarks and turns (Sadalla & Staplin, 1980; Senevirante & Morrall, 1986) and because no objectively "correct" decision exists with respect to trajectory choice, people use implicit strategies to minimize the physical and mental effort involved. As we will see in next subsections, this underlying principle that individual route choice patterns are varied across the population is also considered in the works on trajectory prediction. However, the link to the spatial cognition perspective is barely explicit in any case. In fact, from our knowledge, almost no project aims to understand the driver's reasoning.

The driver route choices can make some specific routes to be imaged as longer or shorter and that fact can be a source of distortions too. In 1995, Golledge tried to understand the criteria most often used in route selection. Thirty-two subjects were selected and experiments were performed both in laboratory and real environment. In the end, the results showed that the top three criteria were shortest distance, least time and fewest turns. These results represent important clues for our work, since the better we can understand the drivers reasoning, the better we will simulate and understand the *Cognitive Maps.* From our intuition, understanding how the driver's map of a city is distorted in time is fundamental to represent properly his/her behaviour model and perform accurate predictions, particularly when facing new routes, as we will see in the next section where our approach is presented.

Route Selection

In the previously subsection we saw that most of the drivers tend to choose the shortest routes to reach their destinations. Based on this fact, our idea consists in distorting the original city graph (whose edge weights represent travel distance) for modeling a *Cognitive Map*, and then using a simple shortest path algorithm (Dijkstra, in our case) to find an optimal route from a cognitive perspective. This graph distortion approach was already used in some related works, but all with different purposes.

Duckham and Kulik (2003) used this procedure to generate the *simplest* route to a destination, in other words, the route that is easiest to explain, understand and memorize. In order to minimize the complexity of a route description, each pair of connected edges has an assigned weight that represents the amount of information required to negotiate their intersection. If the *decision point* is complex (e.g. T- junction) the weight will have a high value, otherwise its value will be lower. After distorting the graph, they simply apply a shortest path algorithm and the *simplest* route is returned.

The same methodology is used in the works presented by Haque et al. (2006) and Caduff et al. (2005). The former was inspired by Duckham and Kulik's work (2003) and it only differs in the weighting function (in Duckham and Kulik's work the weight reflects the instruction complexity of a decision point and in Haque's the weight describes the ambiguity of each turn at an intersection point). In the latter, each node of the graph has nearby landmarks associated with it. In this case, the weight of each pair of connected edges will depend on those landmarks' saliency. The distance/orientation of a driver with respect to them is also

taken into consideration. In the end a revised version of Dijkstra is applied to the graph, returning the *clearest* route in terms of spatial references.

Trajectory Prediction

Trajectory prediction is a large research field that attempts to predict future trajectories of moving objects in several environments. Before taking a deep look into our approach, we should first analyze some important works in the field, specially the ones that focus on the driver's context.

Karbassi and Barth (2003) attempted to predict the route between two stations in a shared-use vehicle's system. They collected GPS traces during three years in order to create route frequency histograms between all pairs of stations during three periods of the day: morning, noon and afternoon. So, before entering the car, the users of this system should insert both origin and destination stations on the station kiosk touchscreen. Then, during the trip, as new location data were received (GPS positions every 30 seconds), a hierarchical tree data structure was used to recalculate the most probable route for the period of the day, based on the histograms described above and links traversed so far.

In 2007, Kim et al. proposed a similar approach to the one described above. Assuming that a driver started a trip from location S and reached the location C by a certain path, moving towards the final destination D, their algorithm starts by looking at all past trajectories taken from S to C. Then, for each one of them, their paths from C were investigated and the most frequent path (the one whose frequency is higher) that passes by the destination D is returned. The authors were not able to collect real-life trajectories, so they did not conduct performance tests.

The main limitation in the two works described above is the fact that it is impossible to predict a driver's trajectory from the scratch (without acquiring prior knowledge), something that we want our algorithm capable of. Moreover, it should also be able to predict both the trajectory and destination without any user input and for a complete city map, in contrast to Karbassi and Barth's work.

There are also some authors that used pattern recognition techniques to predict future trajectories. Simmons et al. (2006) trained a Hidden Markov Model with past routes and destinations in order to predict the next road segments taken by a specific driver until the destination arrival. They tested their approach on a driver with 46 trips, achieving accuracy often above 98%. However, this is a misleading result, because 95% of the driver's road segments end points were connected to only one other road segment. So, the transition was forced and as there was only one option the prediction had to be correct. Thus, the relevant result in this work is the average of 75% of accuracy obtained when the transitions are unforced, that corresponds to 5% of the total predictions.

A similar approach was used in Krumm's work (2008), where he used a simpler Markov Model (instead of a Hidden Markov model) to predict the chain of road segments that a driver

will next take. In contrast to Simmons's approach, in this work only about 28% of the segment transitions are forced, which makes the results more believable. He shows that the accuracy for predicting the next segments rises with the number of past segments matched and drops the farther the model looks into the future. For instance, 90% of accuracy is obtained when only predicting the next, single road segment, 65% when predicting the next five and only 50% when predicting the next ten segments.

In 2008, Krumm and Froehlich tried to guess a driver's entire route, instead of making road segment predictions like in the last two described works. They gathered GPS driving data from 252 drivers resulting in 2.2 million GPS location points with an average of 15.1 days of worth driving data per user. The authors state that a trip is different from a route, since the trip describes a path through time and space and a route is only a collection of latitude and longitude pairs and lacks the time component.

Their algorithm starts by combining similar past trips into routes. In order to do that, they compare every trip in a driver's trajectory data and store the similarity scores between them in a *trip similarity* matrix. Then, they use a hierarchical clustering technique that recursively combines trips until the lowest score in the matrix exceeds a specific threshold. The similarity score between two trips is calculated by computing the average minimum point-segment distance between them.

Thus, as a trip progresses, the distance between the current trip and all the driver's routes (that were calculated using the clustering method referred above) is updated in another *similarity* matrix. Then, two algorithms can be applied: the *Closest match algorithm* returns an ordered list with the most similar routes to the current trip and the *Threshold match algorithm* returns for each route a confidence measure given the trip's distance so far. The results show that by the end of the first mile the percentage of repeated trips correctly predicted is 30% and that by that point, 70% of the times the correct route is already in the top 10 closest ones (increasing to 80% by the third mile). However, the results also show that by the first day, only 1.5% of the trips were considered repeated. Furthermore, we can observe that even after a month of route collection, 40% of the trips are still over new routes so there will always be a lot of trips that will not have a correct match.

CONSTRUCTING THE DISTORTED GRAPH

The representation of the graph has the highest importance since it constrains the algorithms to be used. For some projects referred above, we could see the use of hierarchical graphs (Zhengh et al., 2009), representative routes (Patel et al., 2006; Krumm and Froehlich, 2008), association rules (Morzy, 2007) and changing the original graph with conditional probabilities for each link (Terada et al., 2006). In this paper, we present two different distorting approaches that both act directly on the network graph, in our case distorting individual link distances (or arc weight) according to their use by the driver. In this context, such distorted graph becomes the representation of the city for that specific driver, changing in the same

rate as the driver changes preferences. Acting on the arc weights is also the natural form to influence route planning algorithms.

First Approach

Each link of the network graph will be updated each time the user traverses it. However, such change (a "discount" on the link length) has to be carefully made in a way that reflects the users preferences. Intuitively, there is much more difference between never traversing and traversing once a link than traversing it 100 and 101 times. Furthermore, the real change to the distance has to be enough to make the associated routes to be preferred over other possibilities, but not so much that it makes it mandatory in trips near by. The definition of the weight for a link is:

$$w = (w_{\max} - w_{\min}) \times f(n_{tw}, N_{tw}) + w_{\min} \quad (1)$$

where w_{max} corresponds to the highest possible value of the link (in principle, the initial value, or the value with no passages). w_{min} is the lowest possible value, defined as:

$$w_{\min} = w_{\max} \times \alpha \quad (2)$$

The function $f(n_{tw}, N_{tw})$ is the *distortion function,* with n_{tw} being the total count of passages during the time window tw and N_{tw} a normalization that consists on the maximum count found in all the links for that time window. The time window should correspond to the last tw days (i.e. the window corresponds to [present − tw, present]) so that older link counts are forgot to avoid a strong bias towards older choices and to keep the system faithful to the latest preferences of the user. The current time window is set up to 3 months to completely cover the minimum time needed for routine statistical validity (of 2 months, following results from Krumm and Froehlich (2008)) plus one month of reinforcement.

$\alpha \in [0,1]$ is the *"drivers reality perception factor",* representing the ability of the driver to perceive the *true* distance of a link (or the true distance effect the link has in routes, influenced also by whatever context involved). The variable n corresponds to the number of times the driver traversed that link, while N is a normalization variable that corresponds to the maximum value of n in the whole network.

In our context, the role of a *distortion function* is to change the link weight by the amount that optimizes future predictions. We are looking for a decay function that affects the graph depending on the number of times a link is preferred. Besides being a decay function, we could find no intuitive or theoretical basis to suggest a specific formula. We can easily understand, however, that the main feature relates to how the slope of decay varies over time (e.g. strong decay in the beginning or in the end, smooth variation of the slope, constant slope, etc.).

We selected six different functions: $exp(-x)$, $1/x$, $1-sin(\pi/2\ x)$, $-log_{10}(x)$, $cos(\pi/2\ x)$ and x (linear decay) where x is the normalized quantity n_{tw}/N_{tw}. In Figure 1, we show the effect of each function in the weight according to the weight update equation (1). We assumed $\alpha = 0.5$, $N = 90$ (a link being visit every day during three months) and $w_{max} = 500$ meters.

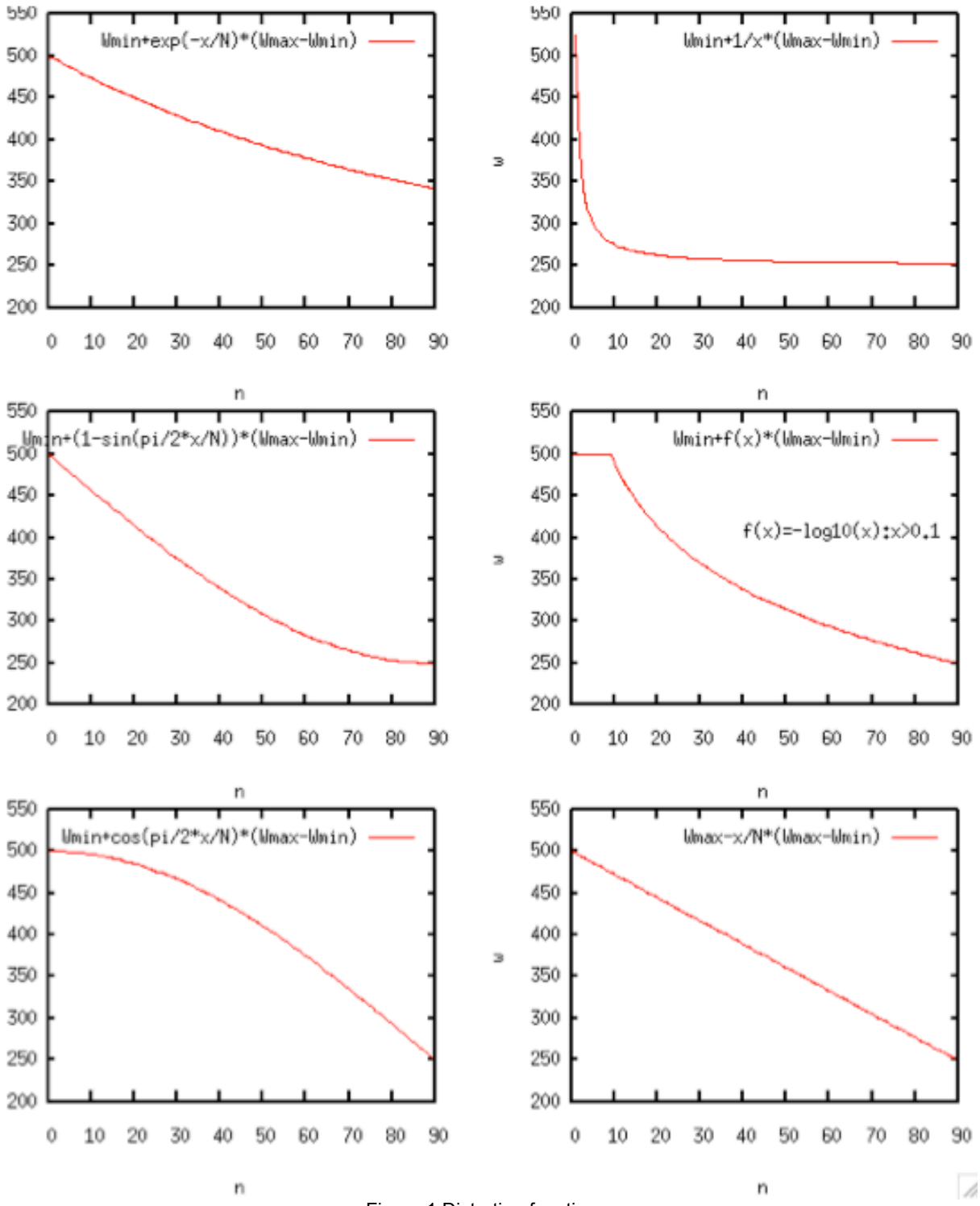

Figure 1 Distortion functions.

Notice the particularities of each function. The decays are different and represent different assumptions about what should be the rate of distortion. For example, using $-log_{10}(x)$, only after an initial number of trips the link is changed, then immediately having the steepest slope (with $n = 10$). This initial ``plateau" is an obvious solution to protect from having an infinite weight with $n = 0$ since it forces the default value of the logarithm to 0. However, this leads to a very abrupt discontinuity at 0.1, so we also add a seventh function where a discontinuity

exists only at 0.01, below which the value will be -2 (notice that -$log_{10}(0.01)$ = -2). In this way, the entire function will have a smooth behavior. This will be called the ``smooth log" function.

With $1/x$ the slope is drastically low during the first visits, abruptly leveled (at $n = 10$) until the end. Cosine and sine are (obviously) complementary, but their subtle difference represents different expectations on the moment of lowest slope (sine in the beginning, cosine in the end). Finally, the linear function has a constant slope (of -(w_{max} - w_{min})/ N). By affecting the graph in different ways, these functions lead to different performances in the trajectory prediction, namely in the number of trips needed to reach accurate predictions.

Second Approach

Let's imagine a scenario where the shortest-path between a driver's work and home is represented by Route A, that has 800 meters of length. However, due to some reason, this driver always chooses Route B (1200 meters) to travel between these two locations, so the algorithm must make Route B shorter than Route A. The idea, differently from the original algorithm, is to decrement Route B and increment Route A in the same proportion every time the driver completes the trip that corresponds to one iteration. The higher the difference between the weights of these two routes the higher should be the *strength* (proportion) applied, so more abrupt will be the distortions. In contrast, the last rectifications will be more accurate, in order to avoid distorting the links more than what is necessary and compromise the map stabilization. Figure 2 illustrates this idea, the *Map Stabilization Effect*.

In order to compute the *strength* value, we start by calculating the difference (in percentage) between the weights of both routes. Then, we simply divide this difference in two equal parts to obtain two equal pieces of strength. One piece will be applied positively in the shortest route (the one we want to increment) and the other one will be applied negatively in the longest route (the one we want to decrement). So, this variable can be defined as:

$$strength = \frac{1 - \frac{a}{b}}{2} \tag{3}$$

where a is the total weight of the shortest route and b the total weight of the longest one. In this way, we will never distort a link more than 49.9(9)% of itself, since the difference between two routes will always be lower than 100%. In the opposite, the lowest possible *strength* value will be 0% that corresponds to those situations where both routes have the same exact length. As we can see in Figure 2 example, the new weights for Route A and B, after the driver passes by Route B twice, are:

1) After one passage (iteration 1):
   - *strength* = $(1 - 800m / 1200m) \div 2 \approx 0.17$ (17 %)
   - *route A* = $800m + (800m \times 0.17) = 936m$
   - *route B* = $1200m - (1200m \times 0.17) = 996m$

2) After two passages (iteration 2):

- *strength* = $(1 - 936m / 996m) \div 2 \approx 0.03$ (3 %)
- *route A* = $936m + (936m \times 0.03) = 964m$
- *route B* = $996m - (996m \times 0.03) = 966m$

In the examples described above we used $f(x) = x$ as distortion function, since the *strength* calculated is directly applied to the links (for instance, $f(17\%) = 17\%$). However, despite of well representing the idea of the *map stabilization effect* (the lower the difference between the weights, the lower the *strength* applied), this function has some limitations.

For instance, the distortion function must not return zero when $x = 0$. If by accident two routes have the exact same length and the driver always chooses the one that is not the shortest for the algorithm, the rectification will never happen, since with *strength* = 0 no distortions will be made. Another limitation is the fact that this function always decreases in the same proportion, which can lead to situations where too small strengths are applied in each iteration (e.g., 0.000001%), making the rectifications impossible to be completed in a valid period.

We believe that there are also other functions that make good representations of the *map stabilization effect* and that are capable of overcoming the limitations described above. Due to its flexibility and since it is commonly used in driver behavior modeling in transport research literature, we decided to try the *Sigmoid* as our distortion function. In our case, this function is defined as:

$$f(x) = \frac{\max val}{1 + e^{-\frac{x - mpoint}{grate}}} \tag{4}$$

where $maxval$ is the maximum value (in percentage) the function can return, $mpoint$ is the x value whose y corresponds to the medium point between zero and $maxval$ and $grate$ corresponds to the slope of the function, in other words, its growth rate. As the function's argument will never be lower than 0% nor higher than 49.9(9)%, we will define the function in $x \in [0,50[$.

The biggest challenge of using this function is to choose the right parameters. We started by defining $maxval = 50$, because we believe that we should not distort an edge more than 50% of its own weight in each iteration. By doing rectifications over that value, we think we would be distorting the links more than what is necessary to rectify them, which could compromise the stability of the map. Instead, we prefer to do smaller adjustments in each iteration and make more accurate rectifications. Moreover, if we want our function to have a similar behaviour to $f(x) = x$ in order to have a good representation of the *map stabilization effect*, defining $f(49.9\%) \approx 50\%$ is a good starting point.

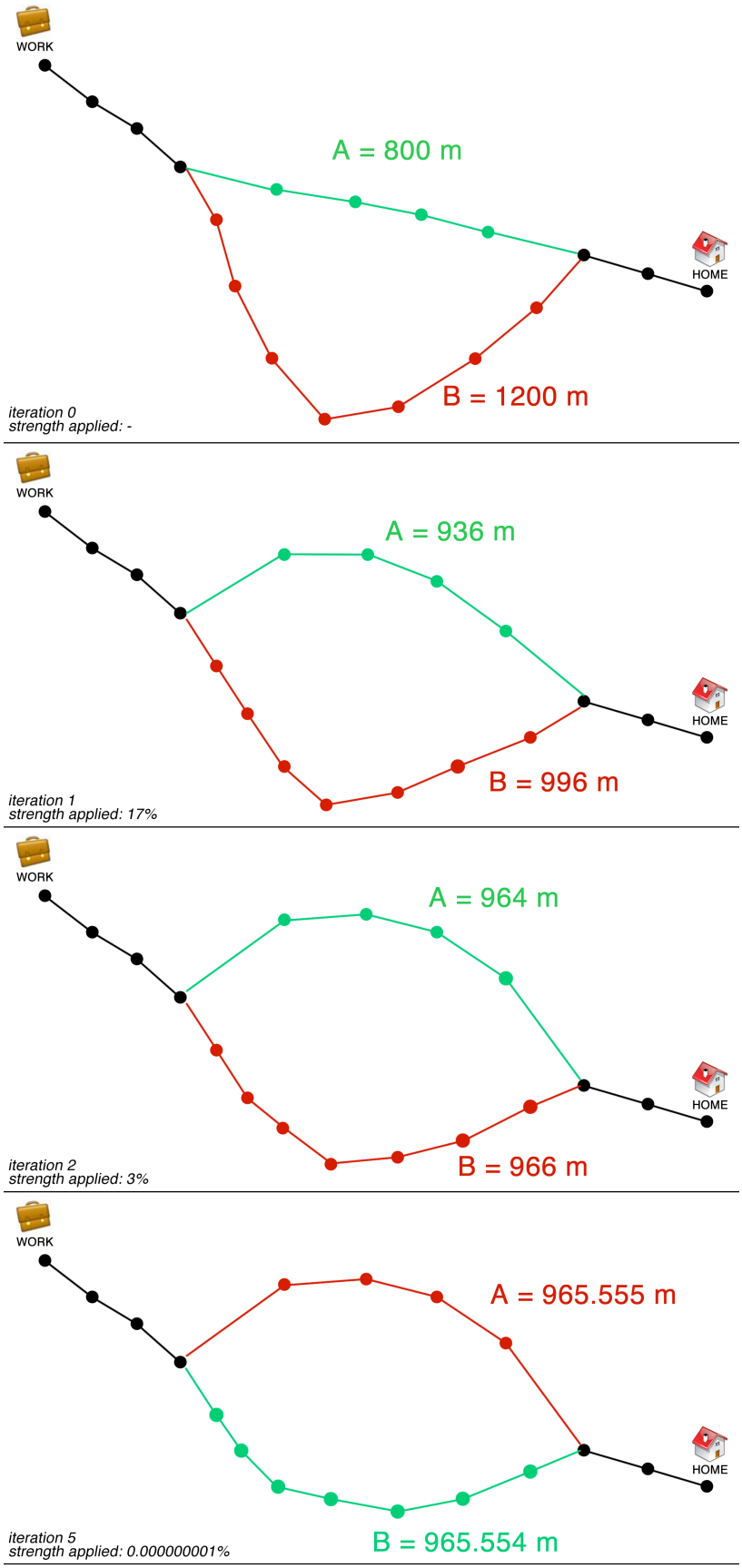

Figure 2 Map stabilization effect.

The remaining parameters are more tricky to choose and difficult to justify. We decided to start our experiments in this approach with $mpoint = 25$, because since we want this function to have the closest possible behavior to $f(x) = x$, this medium point value would guarantee that $f(25) = 25$, and $grate = 5$ because the slope obtained with this value is not too abrupt nor too smooth and we believe it represents a proper learning curve. Moreover, the use of this growth rate implies that $f(0) = 0.3$, which represents the function's absolute minimum. Besides solving the situation of the two same length routes described above, in this way, we also guarantee that the rectifications will be completed in a valid period, since 0.3% is the minimum strength that can be applied in each iteration. We can see the function's appearance in Figure 3.

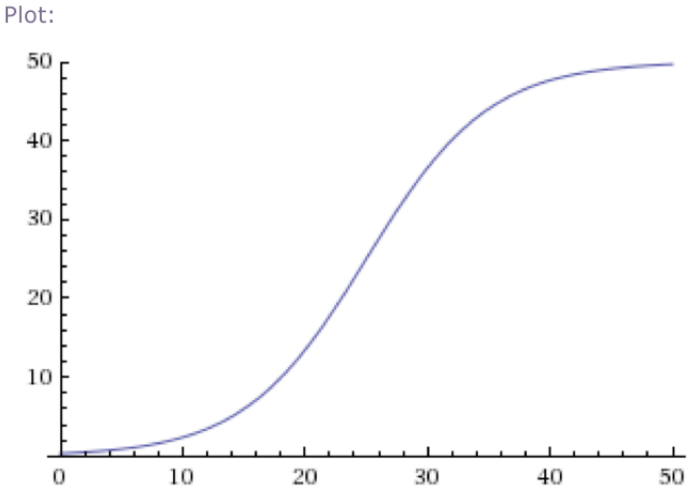

Figure 3 Distortion function's appearance with maxval = 50, mpoint = 25 and grate = 5

By using this function and these parameters, we believe that we are maintaining the *map stabilization effect* represented by $f(x) = x$, but without the limitations referred. However, as it was referred above, these are just our starting values. Only after we start the experiments on this approach with these parameters, we will understand if they are the most suitable for this case or not. So, in that phase it is our intention to understand the influence of the parameters in the performance of the algorithm, in order to adjust their values in the most appropriate way.

DATA PREPARATION

Pre-Processing

As referred earlier, our dataset is composed of eleven different drivers, two from Coimbra (Portugal), one from Seattle (WA, USA) and eight from San Francisco (CA, USA), in a period of one year (from May 2008 to May 2009).

This dataset recorded a broad range of users, outdoor movements, including not only home-work trips but also some extra-routine entertainments. However, it gives no indication of when a trip begins or ends and our algorithm needs knowledge of driver's discrete trips to

yield results. Moreover, some GPS points are noisy and contain invalid sensor data. Thus, before starting experimenting, we performed a series of operations in our dataset.

Bounding box detection: An algorithm was processed for each user's data file in order to understand what was the most appropriate bounding box for each subject. This step is very important because we can not load, for instance, all the map of the state of California to our database (due to system memory problems), so it is crucial to find out which is the bounding box that best fits each driver's traces.

Trip Segmentation: The algorithm used to segment the GPS points in trips looks for gaps between two consecutive recorded points (P1 and P2) of 2 minutes or more. If a gap is found, P1 becomes the end point of the last trip and P2 the beginning point of the current trip. The threshold value of 2 minutes, like some other GPS data processing techniques, was inspired by the work performed by Krumm and Froehlich (2008).

Trip Cleansing: Every GPS trace contains noisy points that represent outliers and our dataset is no exception. So, once the traces were segmented, the GPS points needed to be cleaned. Outliers typically appear as wildly mistaken points along a reasonable sequence of GPS points and normally are far away from the temporally adjacent points in the traces. Thus, like in Krumm and Froehlich's work (2008), we decided to implement a cleansing algorithm based on speed and acceleration to remove these noisy points. We iterate over every GPS point in each trip checking if the segment formed by the current point with the previous one was traversed at more than 200 Km/h and, if so, the current point is removed. With this cleansing we achieved a reduction of approximately 1% of the data points.

Trip Filtering: Due to the excessive size of our dataset, we decided to apply a filtering algorithm that eliminates every GPS point that is less than 10 meters away from the previous one. In this way, we were able to decrease the size of the files in 80%. Also, from Krumm and Froehlich work (2008), we decided to use two filters: the *WithinBoundsTripFilter* and the *MinimumPointCountTripFilter*. The former removes the trips that take place outside the user's bounding box. The latter removes the trips composed of less than 30 GPS points in order to eliminate small and fictitious trips.

Dataset Statistics

The application of these four stages described above to each users data file reduced the number of trips in roughly 73%. Thus, our final dataset is composed of 1892 trips from 11 different subjects. However, due to the "perfectionism" of the map matching algorithm used, on average, each trace was divided in 3.35 smaller ones after the map matching so, the number of trips per driver has increased from 172 to 576 (corresponding to 36 days of worth driving data per user). On average, each one of these traces has 1.1 Kilometers of length and lasted for 4 minutes.

EXPERIMENTS

Prior Prediction

Before starting the experiments on both approaches, we should first be aware of how the algorithm behaves when no distortions are applied to the map graph. So, the purpose of this test is to run every trace of a driver without making edge updates before the beginning of the trips. This test is very important, not just to understand the behavior of the algorithm in this situation but also to further compare its results with the ones yielded by tests that use the distortion function to distort the map in every iteration. Only by comparing both results we can understand if there are any improvements and if it is worth to use the distortion functions or not.

After running this test on all the eleven drivers we obtained, on average, 84% (±5%) of accuracy. This means that, if we just use a simple shortest-path algorithm to predict the route that the drivers will take, without making any distortions on the original city graphs, we can correctly guess, on average, 84% of the road segments by which they will pass over. In fact, in 62% (±7%) of the traces the algorithm was able to correctly predict all the segments, which means 100% of accuracy. These are interesting results because they match with Golledge's studies (1995) that demonstrate shortest distance is the criteria most often used in route selection. With this test, we can see that the base of our approach makes sense. However, our goal is to show that we can improve these results by distorting the city graph according to each driver's past route choices.

First Approach

Preliminary results: The goal of analyzing isolated traces is to understand, in a "controlled environment", how the rate of change affects the quality of the predictions. In so doing, we can contrast the different functions to understand how they affect the evolution of prediction accuracy as the number of trips grows. To understand the minimum number of trips needed to "learn" the users choices is the most relevant question. We focus on the trips that have the lowest accuracies (i.e. the situations where the original map is clearly insufficient) to understand how the distortions help the algorithm to recover from this "worst case scenario".

So, in order to rapidly understand what are the most suitable functions and the most appropriate α, we decided to run a preliminary test on just 30 randomly selected trips that yielded accuracies of less than 50%. This set has cases that range from 20% to 47% of accuracy. For each trip, each function and for four values of (01, 0.3, 0.5, 0.8), we applied the prediction/update cycle for 30 times, recording in each iteration accuracy obtained. In Figure 4, we present the evolution of the accuracy to each of the functions with $\alpha = 0.3$.

We can understand that, for the set of 30 trips, the smooth log and the inverse reach the best results. Another interesting fact is that although the inverse function is quick in obtaining good results, the smooth log outperforms it as the number of iterations grows. Finally, we tested with other values of α, and the patterns are similar, but the lower its value, the faster the

algorithm becomes in reaching higher accuracy. A balance was found at 0.3 that would not be as aggressive to the graph as lower values (e.g. 0.1) but still adapt quickly to new paths. Therefore, we assume the smooth log as our distortion function and $\alpha = 0.3$ for the remaining sections.

Re-substitution Method: The goal of this test is to understand the algorithm's performance when a distortion function is applied to the map edges before the beginning of each trip. So, this test runs every trip of a driver, starting with a non-distorted graph and distorting it in every iteration with the smooth log function and $\alpha = 3$ (training phase). Then, the traces are run a second time, but this time using the graph previously distorted, in opposite to the *Prior Prediction* test where the traces are run using the original one. This is called, in the pattern recognition field, the *Re-substitution Method,* where the train and the test datasets are the same.

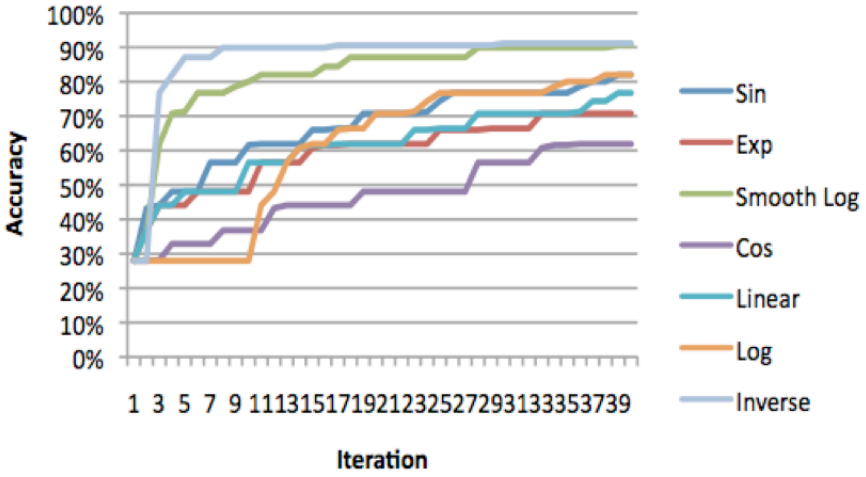

Figure 4 Aggregate (average) behaviour of the distortion functions as a function of n (number of traversals).

In this test we obtained, on average, 90% (±3%) of accuracy using all the eleven drivers as input, which represents improvements of nearly 6% regarding the 84% obtained in the *Prior Prediction* test. In Figure 5 we show how the map distortion affects the accuracy in a typical driver. The X axis of the graph corresponds to all the traces of the driver (in this case, from 1 to 595) while the Y axis represents, for each trace, the difference (in percentage) between the accuracy obtained using this test and the one obtained with the *Prior Prediction* test. Thus, a positive value means that the algorithm was able to correctly predict more segments of a trace using the distorted map instead of the original one. In the opposite, a negative value occurs when the distortions made in the graph deteriorate the accuracy obtained in the *Prior Prediction* test.

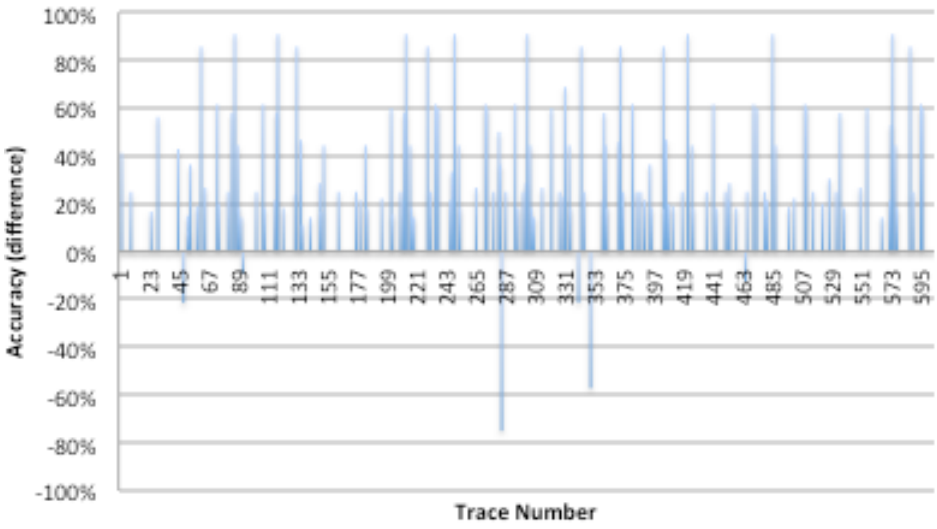

Figure 5 Improvements and degrades caused by the distortion function regarding the *Prior Prediction* test in a typical driver.

As we can see, if we run the traces using a distorted graph instead of using the original one, we obtain much more improvements than degrades. In fact, the use of a distortion function in this driver's map can cause improvements of nearly 90% of accuracy (36% on average), but may also cause degrade in some situations. These cases represent the worst case scenarios, that will be discussed later on this paper.

Second Approach

Finding the optimal parameters: In previous sections we explained why we believed 25 and 5 were the most suitable values for the medium point (M) and growth rate (G), respectively, of the distortion function in this second approach. However, as we were expecting, these values did not produce the best results in the *Re-substitution Method*. By distorting the graph with these parameters, we were only able to obtain 3% of improvements regarding the map with no distortions (*Prior Prediction* test), which is not too much compared with the 6% obtained by the first approach. Thus, our first goal in these experiments was to understand how the parameters affect our distortion function. Since the maximum value of distortion is only a threshold and does not affect the function's appearance, we only studied the influence of the medium point M and growth rate G parameters. Figure 6 shows how the function curve changes with the variation of M and G values.

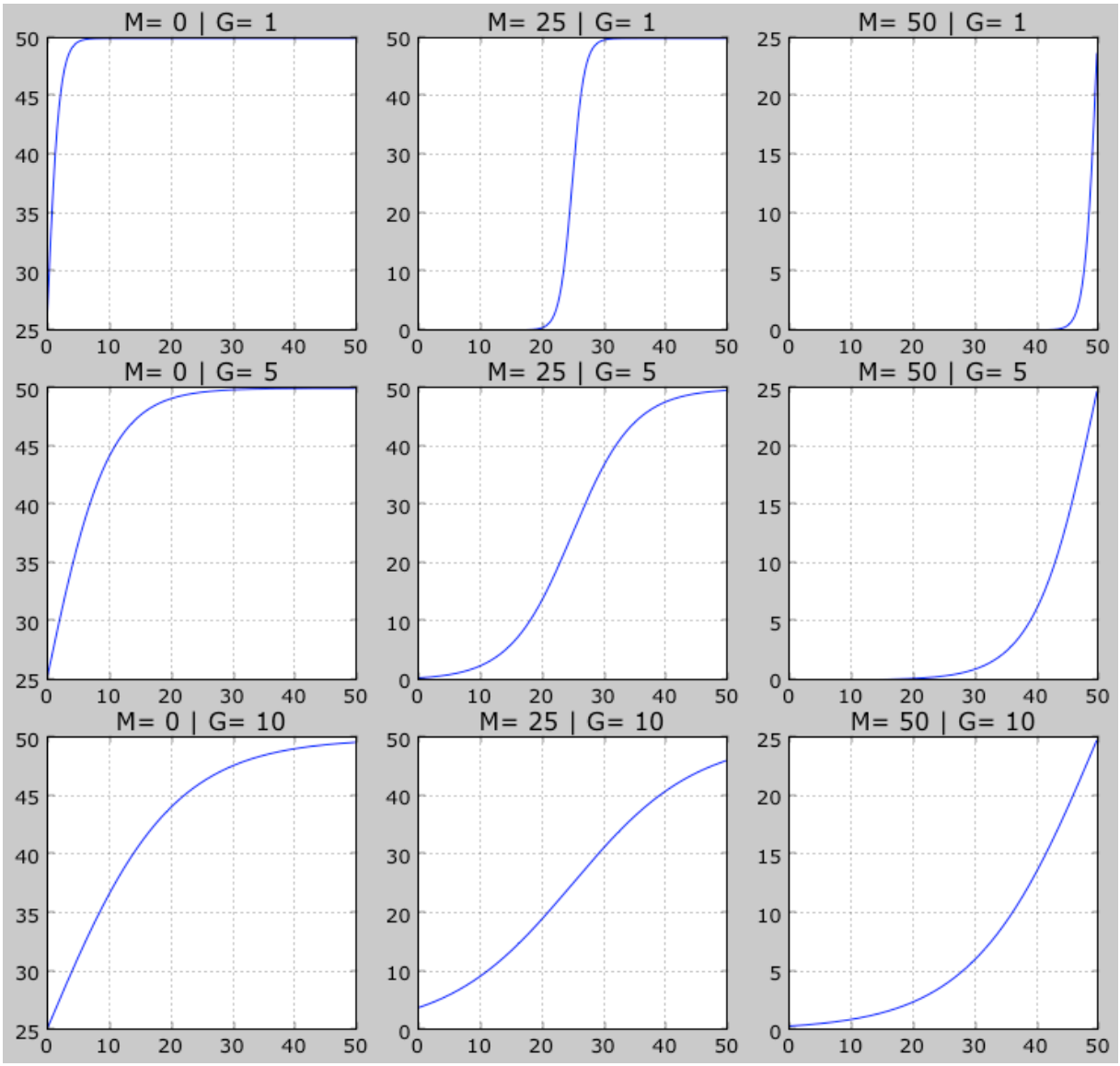

Figure 6 Function flexibility analysis.

As we can see, the lower the growth rate the higher the function's slope and the sooner the function will start to make very small adjustments. In the opposite, a higher *G* makes the learning rate smoother. So, with a very small *G* value (e.g., 2) we are expecting to obtain bad results, since the small adjustments will start too soon and a lot of iterations will be needed until the rectification is completed. For example, let's imagine that there are two different roads (A and B) that end in the same destination: A has 112 meters and B 508 meters of length. For some reason, the driver always chooses route B, so we must make this route shorter than A. If we use our distortion function with the parameters *G=2* and *M=25* we will need 11196 iterations to complete the rectification, in other words, the algorithm will make the correct prediction only after the driver pass 11196 times by that path. However, due to the very small rectifications done in the final iterations (some lower than 0.0001%), the algorithm will correct the paths with a huge precision (A=206.6068m vs. B=206.6065m).

We also believe that a very high *G* (e.g., 70) will cause a bad *FPA* too because, in this case,

the small adjustments will never be made and the function will distort more than what is needed, which can affect traces that share links with the routes that are being distorted. If we maintain the *M* value, change the *G* to 70 and do the exact same test above, we see that we just need 3 iterations to get the weights fixed, due to the abrupt distortions done in each iteration (some over 40%), which means a fast rectification. However, we can also see that the difference between the two routes after the rectification (precision *interval)* is too high (A=222m vs. B=208m). The fact that we are distorting the routes more than what is truly needed (e.g., route A has 222m but could have 206.6068m as we saw above) is not positive at all because some of the traces that share links with, for example, route A may no longer be the shortest ones for the algorithm after this rectification.

The same reasoning is applied to the medium point parameter. If it is too short the consequences are similar to ones verified when the *G* is too high. A very high *M* value has the same effect that a very short *G* value. Thus, we can understand that our *G* and *M* parameters should not be too short or too high, so we need to find a balanced point.

Brute Force Search: In order to find the optimal parameters for our distortion function, we decided to perform a brute force style search. In this search, we vary the *G* value from 0 to 70 and the *M* value from -20 to 50 and perform the Re-substitution Method for all the combinations possible for all the different drivers. Figure 7 shows the accuracy obtained for each combination of *G* and *M* in a typical driver.

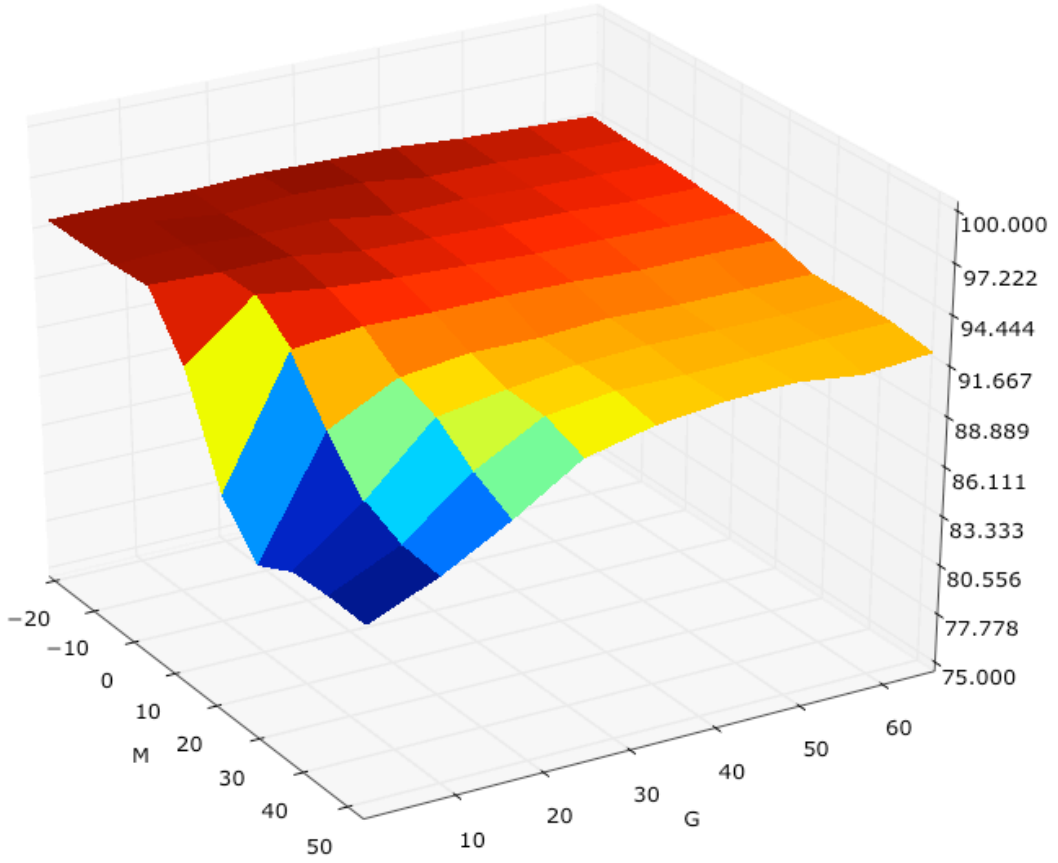

Figure 7 Accuracy (in percentage) obtained for each combination of *G* and *M* in a typical driver using the *Re-substitution Method*.

As we can see, typically, in each driver's graph there is a dark red zone where accuracies slightly above 95% are obtained, which represents improvements of 11% regarding the *Prior Prediction* test and 5% regarding the first approach. This zone corresponds to very low medium points and not too high growth rates. The best results where obtained with *G=9* and *M=-12* (95% ± 2% on average). We can see the function's appearance with these parameters in Figure 8.

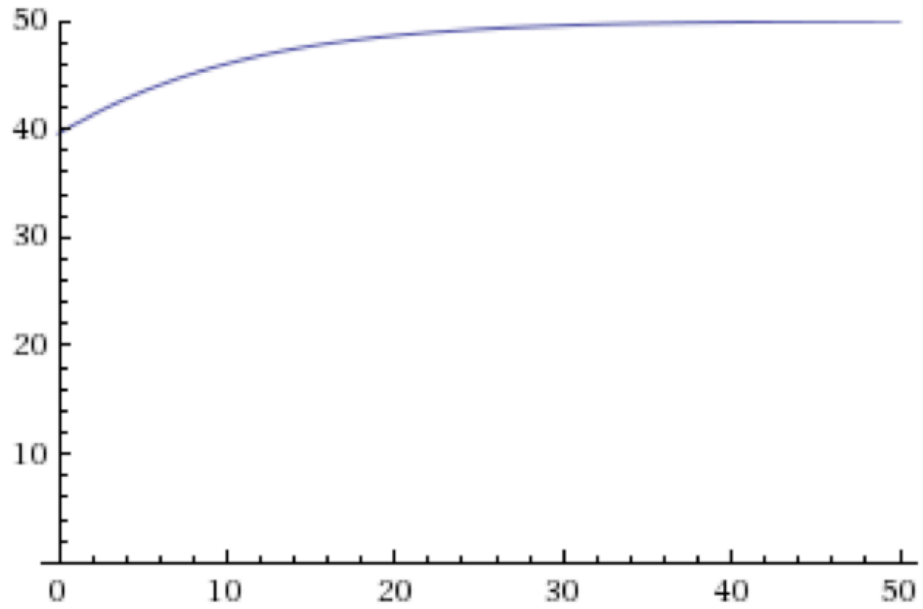

Figure 8 Distortion function's appearance with *G* = 9 and *M* = -12.

The results show that we can get much more improvements if we make strong and abrupt rectifications in each iteration (typically above 40%) instead of trying to represent the *map stabilization effect* idealized initially. In previously sections, we explained why we thought the use of very low medium point was a bad idea. Theoretically, we would get the weights fixed in very little iteration but we would compromise the precision of the rectification. If we repeat the same test performed in the last section, but this time using the parameters that produced the best results, we will see that we just need 2 iterations to fix the weights, but the total difference between them is too high (A=249m vs. B=116m).

After all, a *precision interval* of, for example, 133 meters (249m - 116m) is not bad at all and it is not enough to jeopardize the predictions of traces that share links with the routes that are being distorted. In order to let the reader understand the difficulty of the predictions that are being made, we should say that about 33% (±19%) of the transitions are forced, meaning that there is only one next link for a given link in those cases.

Complete Simulation Test: In order to have an idea of how a simple weight rectification affects the average accuracy, it is important to understand how it changes after each trip is completed. To study those variations, we picked four of the eleven drivers and applied the *Complete Simulation* test, that simulates all a driver's routes in sequence, one route at a time, while checking how both the average accuracy of all his traces and the number of

traces with a bad accuracy (lower than 70%) is changing. Figure 9 shows the graph resulted from applying this test to a typical driver. In this driver, the accuracy starts at nearly 85%, meaning that without making any distortions on the original city graph, we can correctly guess, on average, that percentage of road segments by which he will pass over (Prior *Prediction* test). Then, in general, we can see that the average accuracy increases as the trips are being completed and the map is being distorted, until reaching roughly 95%. It is interesting to see that, although the scales are different, the number of low accuracy traces is inversely proportional to the average accuracy. In this driver, the algorithm was able to reduce the number of these traces from 250 to 88 (from 22% of all the 1144 to 8%), which represents improvements of 14%. In fact, in all the remaining drivers the algorithm was able to reduce the number of low accuracy traces in similar percentages (19% ± 5% on average), which demonstrates a good performance.

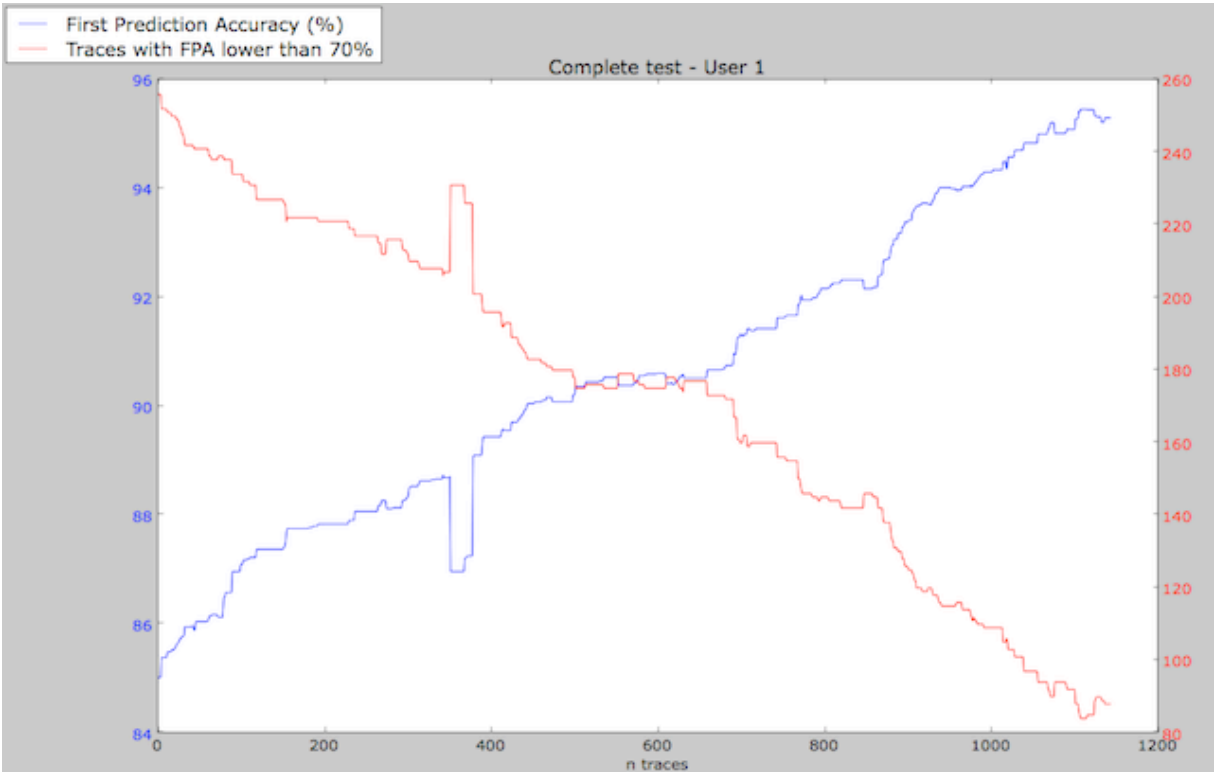

Figure 9 *Complete Simulation* test on a typical driver (driver number 1: 1144 traces).

However, there are still some cases where the rectifications made after a trip cause the decline of the average accuracy. These are the worst cases scenarios, that correspond to those situations where the algorithm is not capable of guessing which path the driver will choose between two points because he usually alternates between both of them. However, every time the driver chooses a different path from the predicted one, the algorithm must proceed with a rectification, which may cause these declines, especially if this new shortest route is only taken sporadically. In these cases, it is impossible for this algorithm to always correctly guess which path the driver will take so, since only one can be the shortest, we will need to add some extra information to our model in the future, like date, time or traffic information.

DISCUSSION

We presented two approaches capable of achieving the goals that were set up: to distort the graph in such a way that the shortest path to the expected destination yields the actual choice of the driver. We chose the tests we thought were the most relevant for this task and we successfully validated the first approach since, on average, it demonstrated a good performance (90% of accuracy) when predicting the routes of eleven drivers from three different environments. This indicates that an abrupt decay may indeed be the best option. We have to acknowledge however that the sample of drivers is highly limited and a wider basis is obviously needed to go further in conclusions. Notice that, in any case, the study of individual traces is independent of the driver (ultimately, we can choose an arbitrary route and check the performance in the same way) being only the full scenario the most sensitive to driver's profiles. Even more important would be the choice of α: the lower its value, the more aggressive is the algorithm in updating distances. Intuitively, a very inefficient set of routes (by a newcomer driver) in comparison to the original map would really need such a low value, while a taxi driver would probably need higher values (knowledge of distance/time should be precise; number of traversals is very high) leading to very smooth changes.

A very important limitation of this study, already acknowledged in this text, is the lack of consideration of context. Of course, people choose different routes according to time of day, day of week, activity (e.g. leisure time, going to work), passengers and weather, to name a few, but even so the algorithm generally performs above our own expectations without external knowledge. Thus, we believe, that a model with several maps according to context (in parallel, in a hierarchical structure) would be sufficient to a real world application.

Another limitation is the dependence on a good destination prediction algorithm. It could even be argued that separating the two is a wrong option in the first place: knowing trajectory so far helps predict destination; destination estimation helps prune the trajectory tree. In fact, our intention is to couple the two pieces together in that way, the full prediction cycle will comprise interactions between the two modules.

Several other minor limitations should also be mentioned. First, the map matching method implemented is a double passage, topological and off-line algorithm, which guarantees high accuracy, but when running online (during a trip), a lighter version may be needed. Second, the creation of many maps to store in memory may become too much consuming and difficult to manage. Another limitation is that a high portion of our daily routine is characterized by high unpredictability, most of it probably impossible to model (e.g. giving a lift to a friend with him guiding the route).

CONCLUSIONS

We presented a study on the process of distorting a road network graph to adapt it to reflect drivers' choices. Such distortion is based on changing link weights (distances) as a function of number of traversals. The rationale is that such graph approximates his/her perception of

space/time. Coupled with the principle that people tend to minimize a cost function (e.g. quickest route) when making a trip, such graph allows for a very simple trajectory (or route) prediction algorithm.

This paper dedicated to design, explain and test a number of parameters for map distortion, namely the distortion function and the α value. The former decides the shape of the decay function while the latter defines the lower limit of weight changes. We apply the trajectory prediction cycle, which consists of running the Dijkstra shortest path algorithm followed by link updates/distortions to minimize the distance between the chosen path and the predicted one. We tested and compared 7 different functions against each other and the "prior" prediction, which simply uses the original map. The comparison between the functions indicates that the logarithm (only truncated near n = 0.01 to prevent from obtaining infinite values, while still providing a smooth descent) and inverse are the ideal ones. The value of 0.3 for α revealed to be the best balance. We also presented a smooth distortion function that deserves to be further explored and tested with more drivers and GPS traces.

Beyond the specific study towards trajectory prediction, the application of the spatial cognition inspiration to individual's mobility choices in general deserves a larger dedication than what is currently found. The estimation of accessibility, for example, a very fundamental topic in Urban and Transport Planning, has been based on crude measures (e.g. gravity model) that rarely pay any attention to the way the individual perceives space.

REFERENCES

Bailenson, J., Shum, M. & Uttal, D. (2000) The initial segment strategy: A heuristic for route selection. *Memory and Cognition, 28(2):306–318*.

Brunyé, T., Mahoney, C., Gardony, A. & Taylor, H. (2010) North is up(hill): Route planning heuristics in real-world environments. *Memory and Cognition, September* 2010, Volume 38, Issue 6, pp 700-712.

Caduff, D., Timpf, S. & Klippel, A. (2005) The landmark spider: Representing landmark knowledge for wayfinding tasks. *In Reasoning with Mental and External Diagrams: Computational Modeling and Spatial Assistance, AAAI Spring Symposium*, pp. 30–35.

Dijkstra, E. (1959) A note on two problems in connexion with graphs. *Numerische Mathematik*, 1.

Duckham, M. & Kulik, L. (2003) "simplest" paths: Automated route selection for navigation. *In COSIT*, pp. 169–185.

Golledge, R. (1995) Path selection and route preference in human navigation: A progress report. *In Frank, A., Kuhn, W., eds.: Spatial Information Theory: A Theoretical Basis for GIS (COSIT '95).* Number 988 *in Lecture Notes in Computer Science, Berlin, Springer,* pp. 207–222.

Haque, S., Kulik, L. & Klippel, A. (2006) Algorithms for reliable navigation and wayfinding. *In Spatial Cognition,* pp. 308–326.

Hirtle, S. & Mascolo, M. (1986) Effect of semantic clustering on the memory of spatial locations. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 12, pp. 182–189.

Karbassi, A. & Barth, M. (2003) Vehicle route prediction and time of arrival estimation techniques for improved transportation system management. *In Intelligent Vehicles Symposium*, 2003, pp. 511–516.

Kim, S., Won, J., Kim, J., Shin, M., Lee, J. & Kim, H. (2007) Path prediction of moving objects on road networks through analyzing past trajectories. *In Knowledge-Based Intelligent Information and Engineering Systems*, pp. 379–389.

Krumm, J. (2008) Markov model for driver turn prediction. *In Society of Automotive Engineers (SAE) World Congress,* April 2008.

Krumm, J. & Froehlich, J. (2008). Route prediction from trip observations. *Society of Automotive Engineers (SAE) World Congress*.

Lloyd, R & Heivly, C. (1987) Systematic distortions in urban cognitive maps. *Annals of the Association of American Geographers*, 77(2):191–207.

Maki, R. (1981) Categorization and distance effects with spa- tial linear orders. *Journal of Experimental Psychology: Human Learning and Memory*, 7, pp. 15–32.

Morzy, M. (2007) Mining frequent trajectories of moving objects for location prediction*. In MLDM '07: Proceedings of the 5th international conference on Machine Learning and Data Mining in Pattern Recognition*, pages 667–680, Berlin, Heidelberg.

Patel, K., Chen, M., Smith, I & Landay, J. (2006) Personalizing routes. In UIST, pages 187–190.

Portugali, J. (1996) The construction of cognitive maps. *Kluwer Academic Publishers*, *Dordrecht*, *Boston*.

Sadalla, E. & Staplin, L. (1980) The Perception of Traversed Distance: Intersections. *Environment and Behavior,* 12(2):167–182.

Senevirante, P. & Morrall, J. (1986) Analysis of factors affecting the choice of route of pedestrians. *Transportaton Planning and Technology*, 10:147–159.

Simmons, R. & Browning, B., Zhang, Y. & Sadekar, V. (2006) Learning to predict driver route and destination intent. *In In Intelligent Transportation Systems Conference, 2006. ITSC 06*. IEEE, pp. 127–132.

Stevens, A. & Coupe, P. (1978) Distortions in judged spatial relation. *Cognitive Psychology,* 10, pp. 422–437.

Terada, T., Miyamae, M., Kishino, Y., Tanaka, K., Nishio, S., Nakagawa, T. & Yamaguchi, Y. (2006) Design of a car navigation system that predicts user destination. *In MDM* '06*: Proceedings of the 7th International Conference on Mobile Data Management, page* 145*, Washington, DC, USA. IEEE Computer Society.*

Tversky, B. (1981) Distortions in memory for maps. *Cognitive Psychology*, 13, pp. 407–433.

Zheng, Y., Zhang, L., Xie, X. & Ma, W. (2009) Mining interesting locations and travel sequences from gps trajectories. *In 18th International World Wide Web Conference (WWW2009), April* 2009.