Collective Intelligence Based Route Recommendation for Assisting Pedestrian Wayfinding in the Era of Web 2.0

Haosheng Huang and Georg Gartner

Institute of Geoinformation and Cartography, Vienna University of Technology, Vienna, Austria

Mobile pedestrian navigation systems are one of the most popular Location Based Services. In the era of Web 2.0, current mobile navigation systems often suffer from the following problems: the lack of social navigation support (utilizing other people's experiences), and the challenge of making user-generated content (UGC) useful. This paper designs some collective intelligence based route recommendation methods to address these problems. The proposed methods can make use of UGC (reflecting other users' navigation experiences), and provide users with the least complex route and the length-complexity-optimized (LCO) route. Some simulations using the street network of the first district of Vienna (Austria) are designed to evaluate the proposed methods. The results show that compared to the shortest route, the collective intelligence based routes have a significant improvement of the route quality (with less complexity rating), thereby more effectively supporting users' navigation tasks (more chances of reaching the destination, fewer mistakes made and shorter distance travelled).

Keywords: user-generated content; collective intelligence; social navigation; route calculation; pedestrian wayfinding

1. Introduction

Mobile pedestrian navigation systems are one of the most important Location Based Services (LBS) (Raper et al. 2007). They aim to effectively assist pedestrians' wayfinding tasks in unfamiliar environments. Recently, mobile pedestrian navigation systems have become increasingly popular not only in citywide outdoor environments but also in many indoor environments, such as shopping malls, complex buildings, and train stations.

Mobile pedestrian navigation systems are designed to facilitate pedestrians' wayfinding. In our daily life, we often ask other more experienced people in the surrounding area for orientation and route advice (i.e. the "social strategy" of wayfinding). These techniques utilizing "experiences of other people" are also known as social navigation

(Dourish and Chalmers 1994). Research has shown that experiences/information from similar users in similar context can help current problem-solvers to gain more efficient and more satisfying answers to their problems (Wexelblat 1999). However, this aspect has widely been ignored in current mobile pedestrian navigation systems.

What's more, in recent years, we have seen a trend towards incorporating Web 2.0's "participation" notion into LBS. These LBS applications employing the "participation" notion enable users to annotate their personal experiences and feelings to physical places during using the systems. The user-annotated information (known as user-generated content (UGC)) reflects the perspectives and experiences of other users who solved their similar spatial tasks in this situation. It can be very useful for current users (e.g. navigators). However, little work has been done on investigating how UGC can be used to generate value/benefits for mobile pedestrian navigation systems.

Recommendation systems can help to make UGC useful. Some examples are "best seller lists" at Amazon.com, "Most viewed" and "Top Favourited" at YouTube, and "Most popular tags" at Flickr, etc. As UGC may reflect other navigators' wayfinding experiences, aggregating UGC can help to provide social navigation support in mobile navigation systems. This paper focuses on designing some route recommendation methods to address the two problems mentioned above. We call these methods collective intelligence based methods because they provide routes by aggregating other navigators' UGC that reflects their collective intelligence (experiences). Two kinds of routes are provided with the proposed methods: the least complex route and the length-complexity-optimized (LCO) route.

The rest of this paper is structured as follows. In section 2, we outline related work. Section 3 describes the proposed collective intelligence based route recommendation methods. Some simulations are designed and implemented to evaluate the proposed methods in section 4. Major results are also discussed in section 4. Finally, section 5 draws the conclusions and presents the future work directions.

2. Related work

The research concerns how mobile pedestrian navigation systems can be improved in the era of Web 2.0. This issue integrates several mainstream trends and concepts, such as mobile pedestrian navigation systems, social navigation, and Web 2.0. From these aspects, we summarize the related work.

2.1 Route calculation in mobile pedestrian navigation systems

Mobile navigation systems often consist of three modules (Huang and Gartner 2009): positioning, route calculation (planning), and route communication. The positioning module determines the current location of a user. For outdoor navigation, GPS is often employed. For indoor navigation, additional installations (e.g. WiFi, Bluetooth, and RFID) are required. The route calculation module focuses on computing the "best" route from an origin to a destination. The route communication module aims to explore technologies to convey route information (directions) efficiently. This paper mainly focuses on route calculation.

For car navigation, fastest routes and shortest routes are very useful and often employed. However, findings in pedestrian behaviour research have shown that pedestrians – especially when having enough time – might favour different route qualities rather than shortness, such as, simplicity, safety, attractiveness, and convenience (Golledge 1995). There is some research focusing on calculating different "best" routes for navigation, such as: routes with a minimal number of turns and routes with a minimal angle (Winter 2002), routes with least instruction complexity (Duckham and Kulik 2003), and reliable routes minimizing the number of complex intersections with turn ambiguities (Haque et al. 2007). However, the above routes are mainly based on the geometrical characteristics of the road network, which may not accurately reflect the qualities (e.g. simplicity, safety, attractiveness, convenience) of the route.

In daily life's wayfinding, we often employ some social strategies, such as asking other more experienced people in the surrounding area for orientation and route advice. This technique is also known as social navigation that is based on the observations that may seem really obvious and simple: Much of the information seeking in everyday life (e.g. choosing where to visit next or which route to take) is performed through watching, following and talking to other people (Höök 2003).The idea of social navigation is often used to help users effectively finding relevant information on the Web, such as the Footprints system (Wexelblat 1999) and the Kalas system (Svensson et al. 2005). It is important to note that very few studies on introducing social navigation technique into mobile pedestrian navigation systems have been done so far. An exception is Gams et al. (2007) which proposed to use other people's trails (i.e. wayfinding experiences) for wayfinding assistance and mainly focused on trail modelling. However, a comprehensive investigation of how social navigation can be used to assist pedestrians' wayfinding is missing.

2.2 LBS in the era of Web 2.0

Recent years have seen a trend towards incorporating Web 2.0's "participation" notion into LBS. These applications, such as Geonotes (Espinoza et al. 2001), E-Graffiti (Burrell et al. 2002), CityFlock (Bilandzic and Foth 2008) and Hycon (Hansen et al. 2004), enable users to annotate their personal experiences and feelings (i.e. UGCs) to physical places or objects during using the systems. Users can also access other users' UGCs. However, these systems can only be viewed as a new form of computer-mediated communication (information exchange) which is location-based.

Volunteered Geographic Information (VGI), termed by Goodchild (2007), is a special case of UGC. Much research on VGI focuses on investigating the behaviour of contributors and their motivation (Budhathoki et al. 2010), data quality (Kounadi 2009), applications of VGI (e.g. disaster risk management) (Poster and Dransch 2010), etc. Recently, the increasing ubiquity of GPS-enabled devices has led to the collection of large spatio-temporal datasets, such as taxi trajectories. A significant number of papers have presented work aiming to mine GPS trajectories of car drivers (e.g. taxi drivers) for route recommendation for car navigation (Letchner et al, 2006, Gonzalez et al. 2007, Ziebart et al. 2008, Yuan et al. 2010). Many of them try to provide routes with less travel time for car drivers.

In this paper, we investigate how UGCs/VGIs can be used to provide route recommendation for pedestrian navigation. Instead of mining GPS trajectories to provide fastest routes, we harness UGCs collected in smart environments to provide pedestrians with least complex routes and length-complexity-optimized routes. UGCs in smart environments reflect the perspective and experiences of other people who solved their spatial tasks in their situation. If most of the other users in similar context consider a specific route as complex, probably, this route can also be viewed as complex for the current user in the current context. From this aspect, other users can help the current user to define/measure route qualities.

3. Collective intelligence based route recommendation

As mentioned before, recommendation methods can help to make UGC useful. It is also a good approach to show the "wisdom of the crowds" (Surowiecki 2004). In this section, we propose two collective intelligence based route recommendation methods, which can make use of UGC from past navigators (reflecting their wayfinding experiences), and provide current navigators with "the least complex route" and "the length-complexity-optimized route".

3.1 Data Modelling

In the UCPNavi (Ubiquitous Cartography for Pedestrian Navigation) project, a smart environment with an indoor positioning module, and a wireless communication module was set up to support navigators' wayfinding, and facilitate navigators' interaction and annotation with the smart environment. Navigators can contribute their UGCs (e.g. ratings, comments, and feedback) to the smart environment during navigating in the smart environment.

For navigation, the route (from an origin to a destination) navigators need to follow can be viewed as route segments connected by decision points (DPs, e.g. street intersections). Making correct decisions at DPs is crucial for successful wayfinding. At different DPs, the complexity of making correct decisions is different.

Users are encouraged to give rating for a DP to indicate their perceived complexity of making the correct decision (choosing the correct road to follow) at this DP. The rating value scales from 1 to 5. The more the complexity, the higher the rating value. A rating for a DP is always involved with a pair of connected route segments (the route segment that the user just visited, and the route segment that the user is going to visit). The current DP is the junction of these two route segments. Therefore, a rating for a DP is modelled as a *5-tuple (user, previous, current, next, rating)* containing a user ID, a previous DP, a current DP, a next DP, and a rating value. It is represented as $R\_DP_{user,previous,current,next}$. For example, in Figure 1, $R\_DP_{u,S,A,B} = 4$ is the rating for DP *A*, and it means that user *u*'s perceived complexity of navigating from *S* to *B* through *A* is 4. It is important to note that for the same navigator, the perceived complexity (effort cost) at a specific DP to its neighbourhoods may be different ("some roads are easy to be recognized"). For example, in Figure 1, user *u* may have the following ratings for DP *A*: $R\_DP_{u,S,A,B} = 4$, $R\_DP_{u,S,A,C} = 2$, $R\_DP_{u,S,A,E} = 5$.

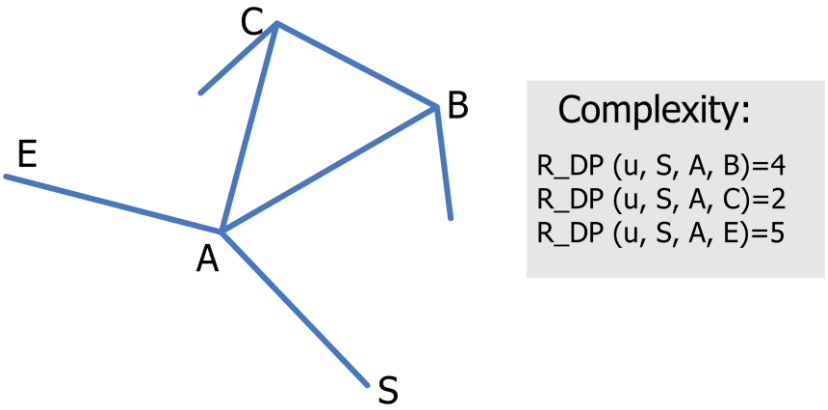

Figure 1. A road network.

In the smart environment, users' moving trajectories may be also employed to unobtrusively infer their implicit ratings for DPs. We do not discuss this aspect here.

3.2 Collective intelligence based route recommendation

Inspired by the "most popular (viewed, discussed)…" like recommendations on the Web, we design some algorithms to provide smart navigation support based on the UGC described in section 3.1. The overall UGC collected in the smart environment reflects users' navigation experiences in the smart environment, and can be viewed as users' collective intelligence. These collective intelligence based algorithms can provide users with the least complex route and the length-complexity-optimized (LCO) route.

It is also important to note that, the collective intelligence based route recommendations for the current navigator are based on the experiences of past navigators (people who have ever navigated in the smart environment before). Current navigator's navigation experiences in the smart environment will be used for assisting future navigators.

3.2.1 The least complex route

The goal of this algorithm is to compute the route with least complexity rating between an origin and a destination. A route from an origin to a destination includes a sequence of DPs. The overall complexity rating of the entire route is simply calculated as the sum of the complexity rating of each DP. As a result, the least complex route can be viewed as route with a minimal overall complexity rating.

For every DP, ratings from different users may be available. We average these ratings to represent the collective intelligence based cost for that DP. To be specific, the collective intelligence based cost (complexity rating) of navigating from node *previous* to node *next* through node *current* (i.e. the current DP), represented as *CI_DP(previous, current, next)*, is calculated as the average of all $R\_DP_{*,previous,current,next}$ ratings.

Dijkstra's algorithm is often used to calculate the shortest (cost) routes from a graph (Dijkstra 1959). The basic idea of Dijkstra's algorithm is to assign some initial distance values and try to improve them step-by-step. Figure 2 depicts the pseudo code of the algorithm.

```
1  function Dijkstra(Graph, origin, destination):
2      for each vertex v in Graph:           // Initializations
3          dist[v] := infinity,    previous[v] := undefined
4      dist[origin] := 0                     // Distance from origin to origin is 0
5      Q := the set of all nodes in Graph // All nodes in the graph are unoptimized - thus are in Q

6      while Q is not empty:                 // The main loop
7          u := vertex in Q with smallest dist[]
8          if dist[u] = infinity:     break                    // There is no route from origin to destination
9          if u = destination:        break                    // reach the destination
10         remove u from Q
11         for each neighbor v of u:         // where v has not yet been removed from Q.
12             alt := dist[u] + cost_between(u, v)
13             if alt < dist[v]:             //If the distance is less than the previously recorded distance
14                 dist[v] := alt,       previous[v] := u

15     //read the shortest path
16     S := empty sequence
17     u := destination
18     while previous[u] is defined:
19         insert u at the beginning of S
20         u := previous[u]

21     return S
```

Figure 2. The pseudo code of Dijkstra's algorithm (adapted from (Wikipedia 2011)).

When introducing cost for pairs of connected edges, the least cost route may include cycles. For example, in Figure 3, the route with a minimal rating from *S* to *E* is *(S, A, B, C, A, E)*, which crosses the centre node *A* twice[1].

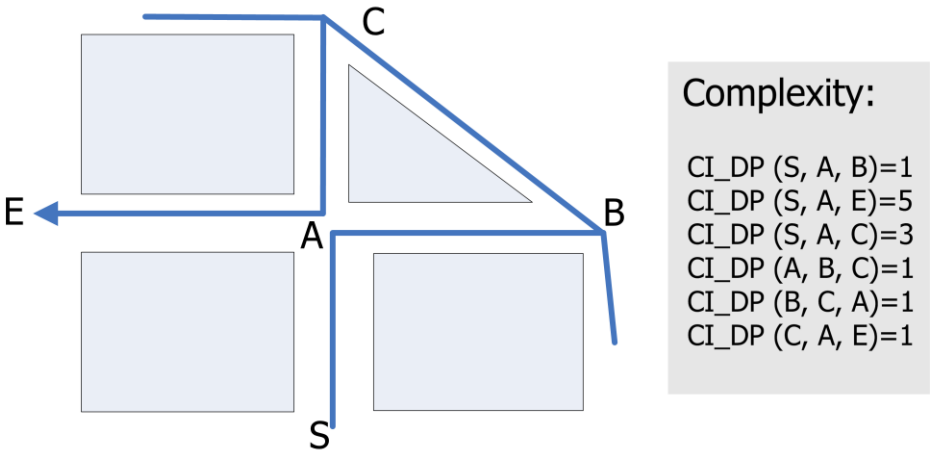

Figure 3. The least complex route crosses the centre node A twice.

Therefore, it is inappropriate to simply adapt the cost function in line 12 of Figure 2 to "*alt:=dist[u]+CI_DP (previous[u], u, v)*", and run Dijkstra's algorithm. Because at the

[1] However, it is important to note that this situation is not likely to happen in the real world.

second step of the main loop (*lines 6-14*) in Figure 2, node *A* will be removed from *Q* (list of unvisited nodes), and not checked anymore. Finally, Dijkstra's algorithm will report that "there is no route from S to E".

To solve this problem, we use the restricted pseudo-dual graph proposed by Winter (2002). The pseudo-dual graph *D* of an original graph *G* is defined as: 1) Each edge e_i of *G* is represented as a node v_i in *D*, 2) Each pair of connected edges (e_i, e_j) in *G* is represented as an edge ε connecting nodes v_i and v_j in *D*. Winter (2002) proves that the shortest (cost) route (single-source/single-target) problem in the original graph *G* can be transformed to a multi-sources/multi-targets problem in *D*. He reduces this problem to a single-source/single-target problem by adding a virtual source node and a virtual target node to *D*. In this new graph *D'*, the shortest (cost) route can be computed by using Dijkstra's algorithm.

For our case, a collective intelligence based cost for a DP $CI\_DP_{A,B,C}$ can be easily assigned to the corresponding edge in the pseudo-dual graph *D'* :

$$cost(v1,v2) = CI\_DP(A, B, C) \quad (1)$$

Where v_1 in *D'* is the edge *(A, B)* in *G*, and v_2 in *D'* is the edge *(B, C)* in *G*.

Based on *D'*, we adapt line 12 of Figure 2 to "*alt:=dist[u]+cost(u,v)*", and use Dijkstra's algorithm (Figure 2) to compute the route with least complexity between an origin and a destination. The result of this calculation is the least complex route in graph *D'*. It can be easily transformed back to the route in graph *G*.

3.2.2 The length-complexity-optimized route (the LCO route)

Compared to the shortest (distance) route, the least complex route in section 3.2.1 may lead to a longer distance between an origin and a destination. Literature has shown that distance is one of the most important criteria for route choice (Golledge 1995). As a result, we calculate the LCO route, which considers both complexity ratings of DPs, and the Euclidean distance of route segments. In order to calculate the LCO route, we assign some weights to the complexity rating of DP, and the Euclidean distance. This optimum cost is given by:

$$CI\_Optimal(A, B, C) = \lambda * CI\_DP(A, B, C) + (1-\lambda) * Dist(B, C) \quad (2)$$

Where *Dist(B, C)* is the Euclidean distance of route segment *(B, C)*, and *CI_DP (A, B, C)* is calculated as the method in section 3.2.1. $\lambda \in (0,1)$ is the weight.

Similar to what we did when calculating the least complex route, we also represent these optimum costs in the pseudo-dual graph *D'*. In addition, for every node *v* in *D'* (i.e. an edge in the original graph *G*), we assign the Euclidean distance of the corresponding edge as its cost *Len(v)*.

As a result, based on *D'*, we adapt line 12 of Figure 2 to "*alt:=dist[u]+ λ *cost(u,v)+ (1-λ) *Len(v)*", and use Dijkstra's algorithm (Figure 2) to compute the LCO route between an origin and a destination.

For every smart environment, the optimum value for λ needs be learned. Section 4.3 gives a detailed description on how to learn the value.

4. Simulations and results

In this section, we design some evaluations for the proposed methods. The setting and the metrics of the experiments are discussed in section 4.1. The results are presented in section 4.2 (the least complex route) and section 4.3 (the LCO route). Section 4.4 summarizes the findings of the experiments.

4.1 Experiment setting

The proposed routes and human performance when following these routes should be evaluated in an experimental manner. Because of the high cost of evaluating the algorithms with human participants (large-scale field studies are needed in order to provide a meaningful evaluation), some simulations are designed. Similar to Haque et al. (2007), two kinds of evaluations are performed.

4.1.1 Setting of the first evaluation

The first evaluation analyses the least complex route and the LCO route by considering length and complexity rating. Its corresponding shortest (distance) route (i.e. from the same origin to the same destination) using Dijkstra's Algorithm is also calculated as a benchmark. The street network of the first district of Vienna, Austria is used as the study area.

At a DP (e.g. a street intersection), the number of available alternatives may affect navigators' perceived complexity of choosing a correct road[2].The more outgoing alternatives at a DP, the more complex people might feel to choose the correct road. Navigators are more

[2] Some other factors may also affect navigators' perceived complexity of a DP, such as spatial layout, signage, differentiation and whether there are landmarks at the DP (Montello 2005).

likely to give higher complexity rating for a DP that has more outgoing alternatives. As a result, the rating a navigator gives to indicate the complexity of a turn from edge *(S,A)* to edge *(A,B)* is simulated by: $R\_DP_{u,S,A,B} = (N_A\text{-}2)/(N_A\text{-}1)$, where N_A is the number of branches at the decision point *A*.

For any two nodes (not directly connected) in the street network, the proposed methods and Dijkstra's Algorithm are employed to calculate the least complex route, the LCO route and the shortest route respectively.

4.1.2 Setting of the second evaluation

The second evaluation simulates human navigators traversing these routes, and compares the performance (of navigators) when following the least complex route, the LCO route and its corresponding shortest route. The following assumptions are set for the simulated navigator (agent):

(1) If it is told to choose a road at a DP (on the calculated route), it would choose one of the outgoing alternatives of this DP (i.e. it does not go back).
(2) In case the navigator makes a wrong choice at a DP and subsequently arrives at another intersection (deviating from the calculated route), it would realize that it has made a mistake[3], and goes back to the previous DP.
(3) If the simulated navigator has made 10 errors during following the route, it stops. We consider it fails to reach its destination.

Similar to Haque et al. (2007), the following metrics are employed when comparing the performance of navigators using the shortest route with that of the least complex route, and with that of the LCO route:

- **Failed travel:** If the navigator has made 10 errors during following the route, its travel is considered as a failed travel
- **Total distance travelled when successfully arriving at the destination:** the total distance travelled by the simulated navigator following the calculated route (including distance travelled due to errors)
- **The number of errors when successfully arriving at the destination:** the number of times the simulated navigator realizes that it is on the wrong route, and goes back to the previous DP

[3] In the real world, a human navigator can be reminded by the navigation system that she/he has made a mistake.

- **Distance of stopping point to the destination when failing to reach the destination:** the shortest distance between the place where the simulated navigator finally stops (after 10 errors) and the destination. With this, we want to know whether the proposed routes can bring navigators closer to the destination even if they fail to reach the destination.

4.2 Results: the least complex route

The street network dataset of the first district of Vienna (Austria) is employed. More than 135,000 triples of routes (with the length longer than 300 metres, involving at least 1 turn) are calculated using the proposed methods and Dijkstra's Algorithm.

Figure 4 shows a typical example of a least complex route and its corresponding shortest route. The least complex route is only 17% longer than the shortest route, while with 38% less complexity (in rating) than the shortest route.

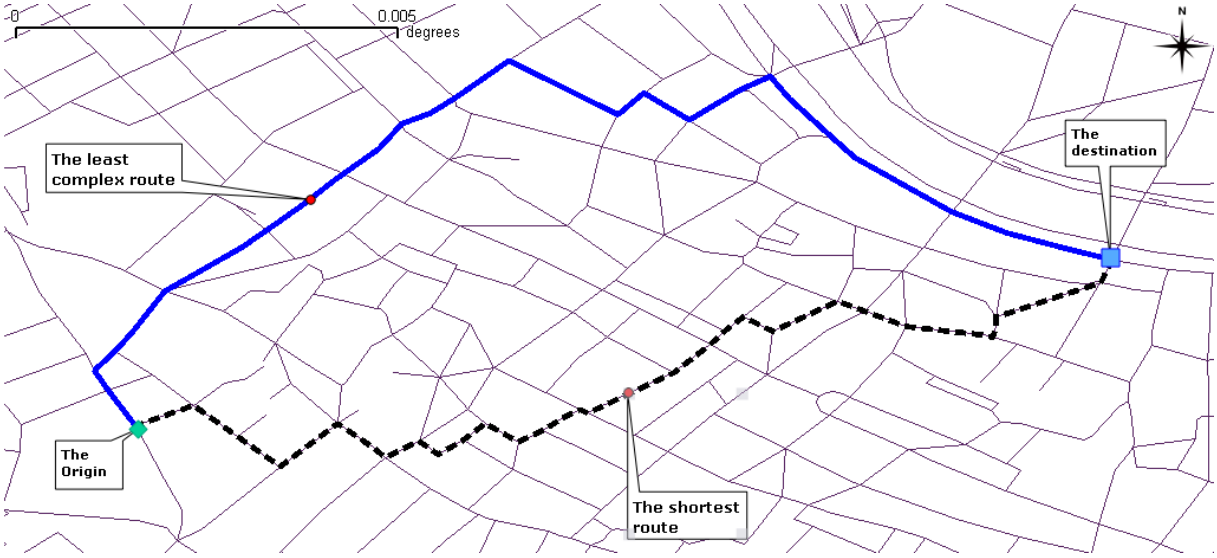

Figure 4. A typical example: comparison of a least complex route and its corresponding shortest route, complexity rating (36 vs. 58) and distance (1472 metres vs. 1259 metres).

4.2.1 Results of the first evaluation: the least complex route

Figure 5 depicts the comparison in terms of complexity rating. The difference in complexity rating is measured as $C_{shortest} - C_{least\_complex}$, where $C_{shortest}$ and $C_{least\_complex}$ are the complexity ratings of the shortest route and the least complex route respectively. It shows that for all the pairs of routes, the least complex route has a lower complexity rating than its corresponding shortest route. The average difference in complexity rating between the shortest routes and the

least complex routes is about 3.68. With the increase of the distance between origin and destination, the difference in complexity rating increases.

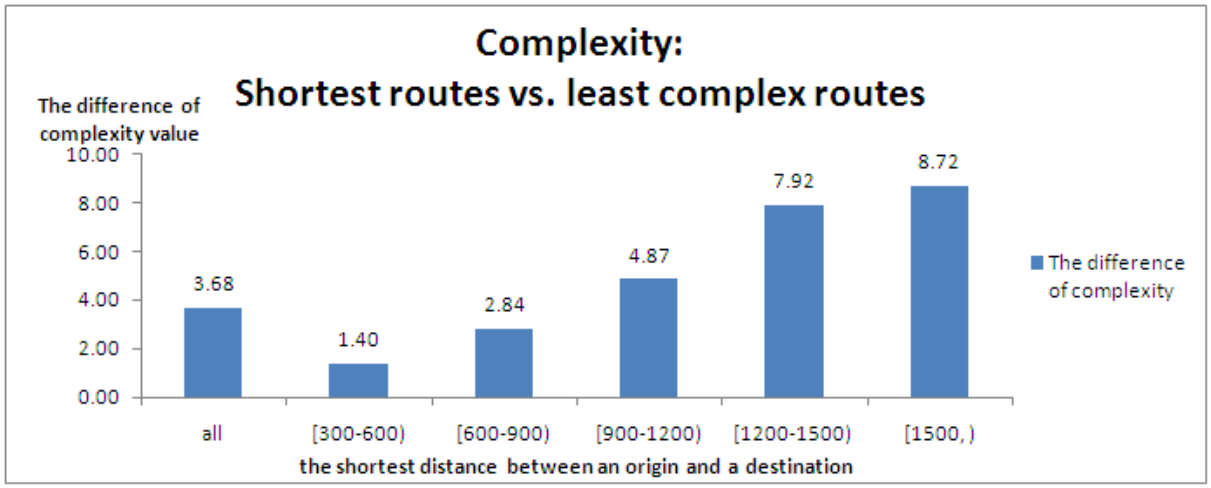

Figure 5. Comparison of the complexity rating of the least complex route and its corresponding shortest route.

Figure 6 shows how the increase of length (in percentage) changes with the decrease of complexity rating (in percentage) among different pairs of origin and destination with various distances. The increase of length (in percentage) is calculated with ($L_{least\_complex}$ - $L_{shortest}$)/ $L_{shortest}$ * 100%, and the decrease of complexity rating (in percentage) is measured with ($C_{shortest}$ - $C_{least\_complex}$)/ $C_{shortest}$ * 100%, where $C_{shortest}$, $L_{shortest}$, $C_{least\_complex}$ and $L_{least\_complex}$ are the complexity ratings and the lengths of the shortest route and the least complex route respectively.

The results show that the least complex routes are on average 7.84% longer than the shortest routes, while on average with 9.92% less complexity (in rating) than the shortest routes. The gaps (between increase of distance in percentage and decrease of complexity rating in percentage) widen with the increase of the distance between origin and destination.

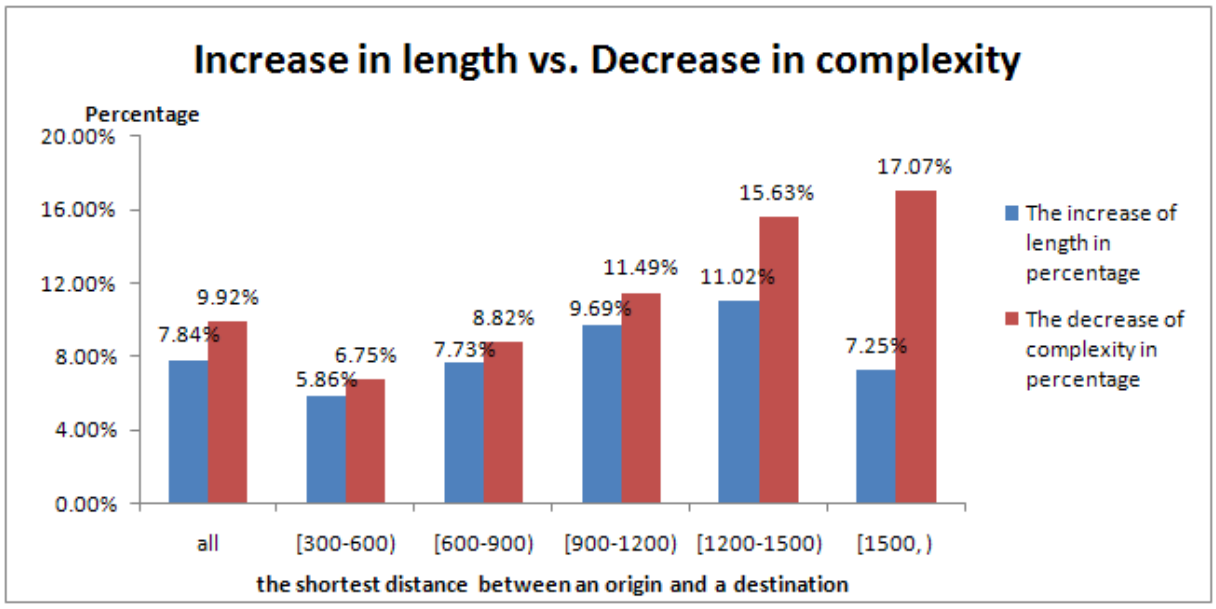

Figure 6. How the increase (in percentage) of length changes with the decrease (in percentage) of complexity rating among different pairs of origin and destination with various distances. The least complex routes are on average 7.84% longer than the shortest routes, while on average with 9.92% less complexity than the shortest routes.

In summary, Figure 5 and 6 confirm our expectation that the least complex routes lead to improvement of the route qualities (with a less complexity rating).

4.2.2 Results of the second evaluation: performance of navigators using the least complex route

To compare the performance of navigators following the least complex route and its corresponding shortest route, we run the simulated navigator 100 times per route (in total we have 135817*3*100 simulations), and use the average value for each metric described in section 4.1.2 per route.

Figure 7 compares the shortest route and the least complex route in terms of the percentage of failed travels. The bar graph shows that the average percentage of failed travels using the shortest route is considerably higher than the percentage of failed travels using the least complex route (19.23% vs. 7.13%). For all different pairs of origin and destination with various distances (except the distance shorter than 600 metres), the percentage of failed travels using the shortest route is higher than that of using its corresponding least complex route.

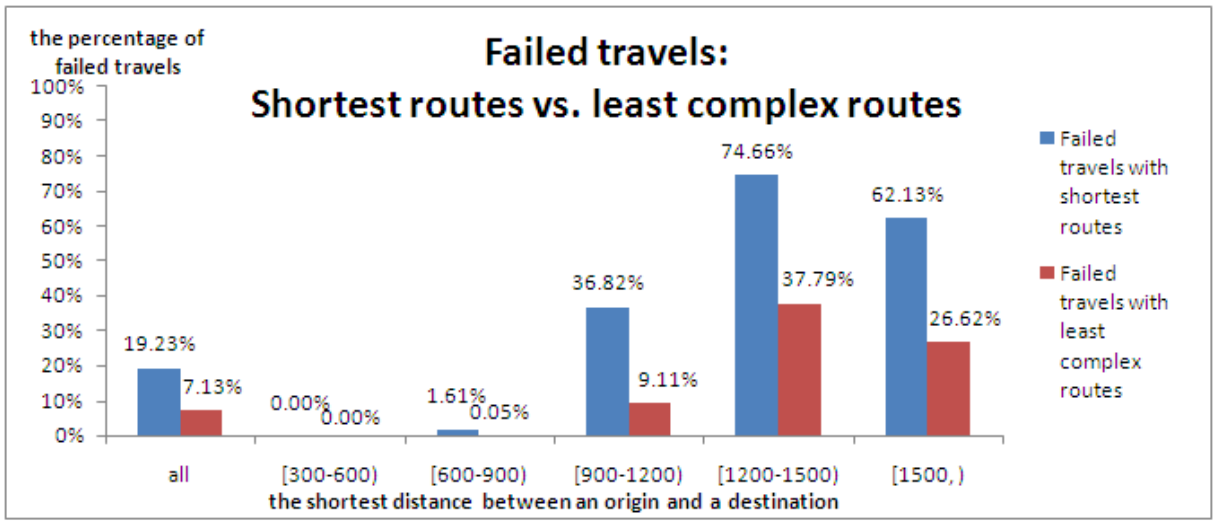

Figure 7. Comparison of the percentages of failed travels when using the shortest routes and the least complex routes. Using the least complex routes raises the chance of reaching a destination.

Figure 8 depicts the comparison of total distance travelled (left) and the number of errors made (right) when the simulated navigator reaches a destination. The bar graphs show that the total distance travelled using the shortest routes is bigger than that of its corresponding least complex routes. Similar results about the number of errors can be found.

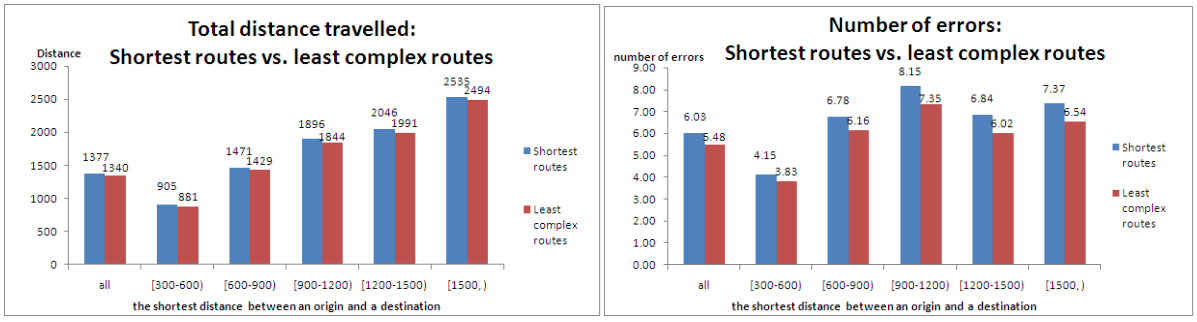

Figure 8. Total distance travelled in metres (left) and number of errors (right) when the simulated navigator reaches a destination, comparing the least complex routes and the shortest routes.

Figure 9 depicts the distance between the place where the simulated navigator finally stops (after 10 errors) and the destination. From the results, there is a clear advantage for the least complex route in terms of distance of stopping point to the destination.

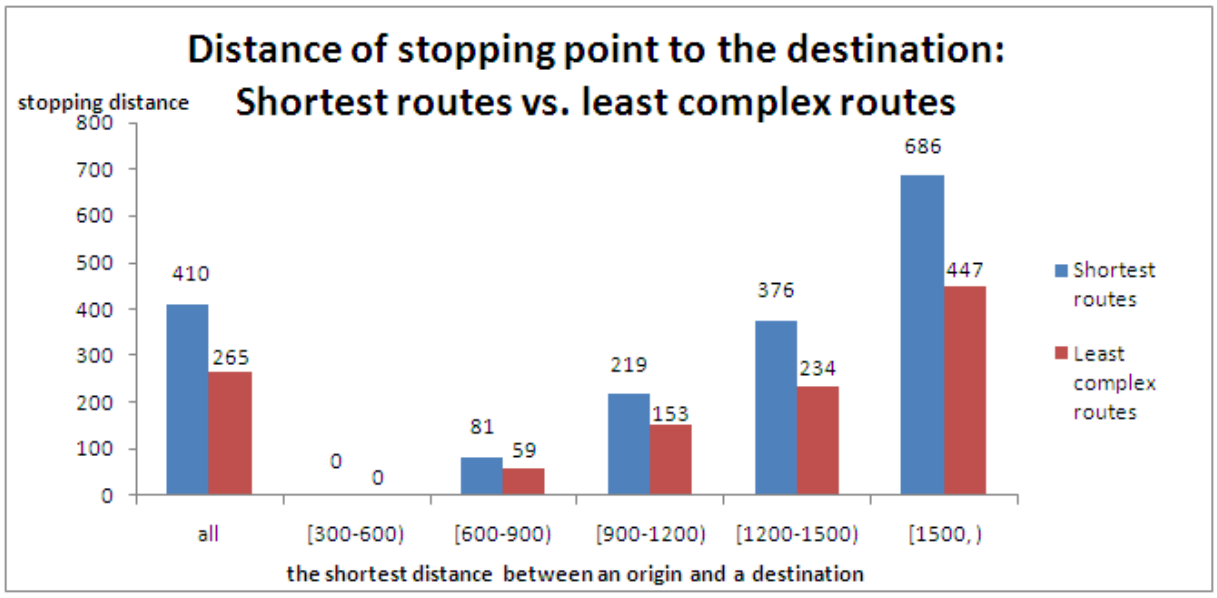

Figure 9. Distance (in metres) of stopping point to destination, comparing the least complex routes and the shortest routes.

In summary, when following a least complex route, the simulated navigator reaches the destination more often than when traversing the corresponding shortest route. For all successful travels, simulated navigators using the least complex routes make fewer errors, and travel shorter distance. For all failed travels, simulated navigators following the least complex routes get closer to the destination than navigators using the corresponding shortest routes.

4.3 Results: The length-complexity-optimized route (the LCO route)

It is clear that the least complex route has a lower complexity rating, but it also leads to a longer distance between two locations (see Figure 6). The LCO route tries to find an optimal trade-off between the least complex route and the shortest route.

To estimate the optimum value for λ, we evaluate several hundred values, and use the best-performing one for our final simulations. The evaluation considers the difference in complexity rating between the least complex route and the LCO route, and the difference in length between the shortest route and the LCO route. The goal is to find value for λ that best approximates both, complexity rating and distance. Finally, λ is optimized as $20/21 \approx 0.95$.

Figure 10 shows an example of a shortest route, a least complex route and a LCO route. The LCO route is only 5% longer than the shortest route, and 17% shorter than the least complex route, while with 14% less complexity (in rating) than the shortest route, and with 28% more complexity (in rating) than the least complex route.

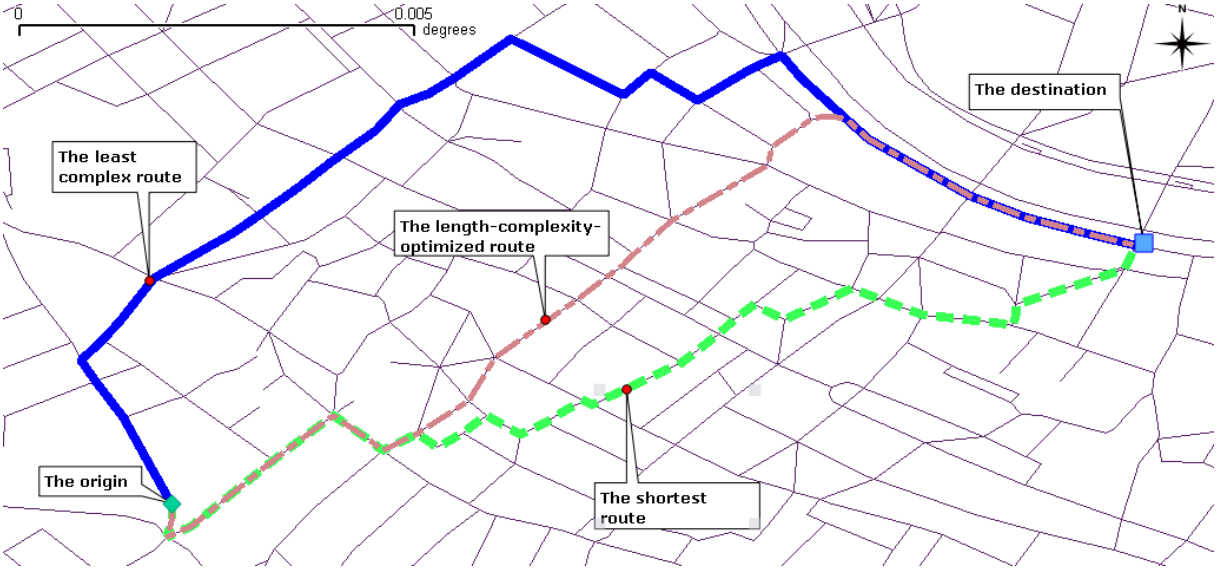

Figure 10. A typical example: comparison of a shortest route, a least complex route and a LCO route, complexity rating (57 vs. 38 vs. 49) and distance (1265 metres vs. 1603 metres vs. 1333 metres).

4.3.1 Results of the first evaluation: the length-complexity-optimized route (the LCO route)

Figure 11 shows the decrease (in percentage) of complexity rating and increase (in percentage) of length when comparing the LCO route and the least complex route. The increase of length in percentage is calculated with $(L_{least\_complex} - L_{shortest})/ L_{shortest} * 100\%$, and $(L_{optimized} - L_{shortest})/ L_{shortest} * 100\%$. The decrease of complexity rating in percentage is measured with $(C_{shortest} - C_{least\_complex})/ C_{shortest} * 100\%$, and $(C_{shortest} - C_{optimized})/ C_{shortest} * 100\%$. The results show that the LCO routes are on average 2.08% longer than its corresponding shortest routes, while on average with 7.54% less complexity (in rating) than its corresponding shortest routes.

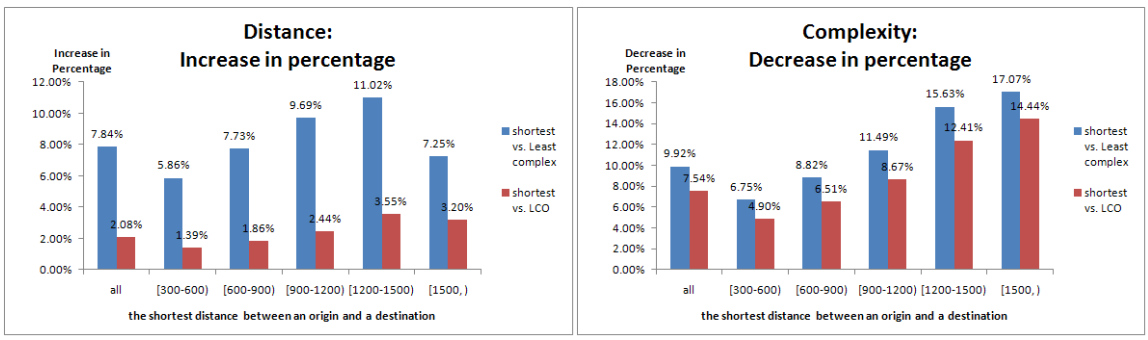

Figure 11. The decrease (in percentage) of complexity rating (left) and increase (in percentage) of length (right).

In order to make a clear conclusion, we also compare the gaps (between increase of distance in percentage and decrease of complexity rating in percentage) between shortest-LCO and shortest-least_complex. Figure 12 depicts the results. It shows that the gaps of shortest-LCO are considerably bigger than that of shortest-least_complex. It means that the LCO routes can achieve a better trade-off between distance and complexity rating than the least complex routes.

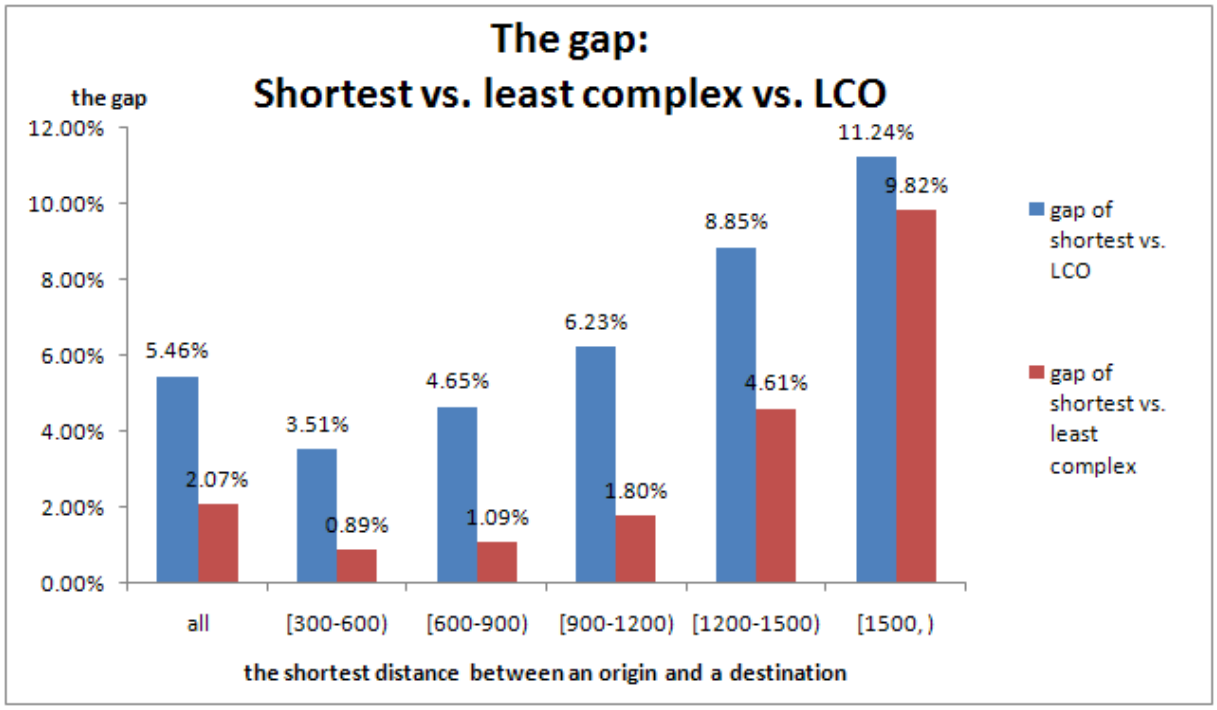

Figure 12. Comparison of the gaps (between increase of distance in percentage and decrease of complexity rating in percentage) between shortest-LCO and shortest-least_complex. The gaps of shortest-LCO are considerably bigger.

In summary, the LCO routes provide a better trade-off between complexity rating and distance than the other two kinds of routes.

4.3.2 Results of the second evaluation: performance of navigators using the LCO route

Figure 13 compares the shortest route, the least complex route and the LCO route in terms of the percentage of failed travels. The bar graph shows the least complex routes achieve the best performance in terms of the percentage of failed travels, followed by the LCO routes, and the shortest routes (7.13% vs. 15.70% vs. 19.23% on average). For different pairs of origin and destination with various distances, similar results can be found.

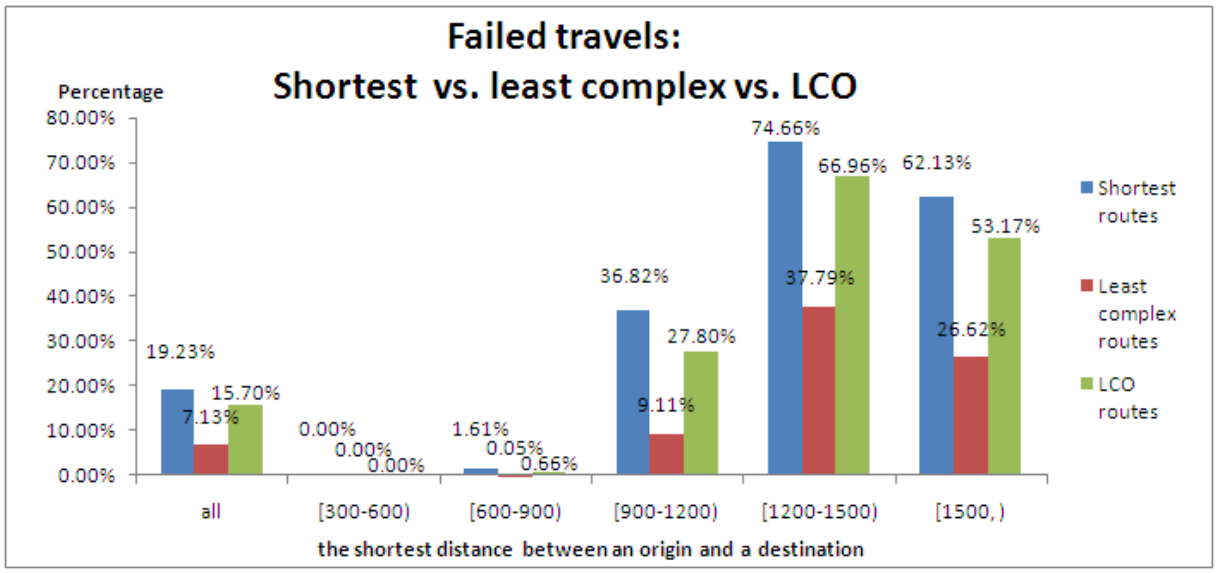

Figure 13. Comparison of the percentages of failed travels when using the shortest routes, the least complex routes and the LCO routes.

Figure 14 depicts the comparison of the number of errors made and total distance travelled when the navigator reaches a destination. Similar to the results in Figure 13, the least complex routes achieve the best performance in terms of the numbers of errors, followed by the LCO routes, and the shortest routes (5.48 vs. 5.61 vs. 6.03 on average).

It is important to note that in terms of total distance travelled, the LCO routes achieve the best performance, following by the least complex routes, and the shortest routes (1330 metres vs. 1340 metres vs. 1377 metres on average). It means that simulated navigators using the LCO routes travels less distance compared to navigators using the other two kinds of routes.

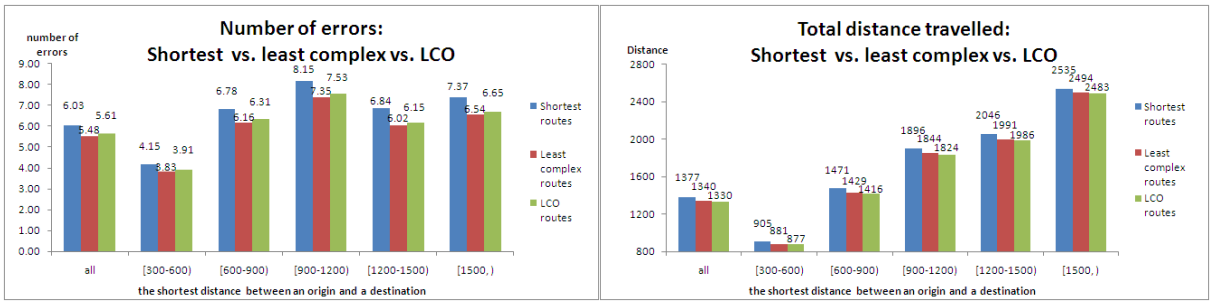

Figure 14. The number of errors (left) and total distance travelled in metres (right) when the navigator reaches the destination. Be aware of the right bar graph, it shows that simulated navigators using the LCO routes travel less distance compared to navigators using the other two kinds of routes.

Figure 15 depicts the distance between the place where the simulated navigator finally stops (after 10 errors) and the destination. From the results, there is a clear advantage for the

least complex route and the LCO route in terms of stopping distance to the destination. The difference in distance between the least complex route and the LCO route is considerably smaller.

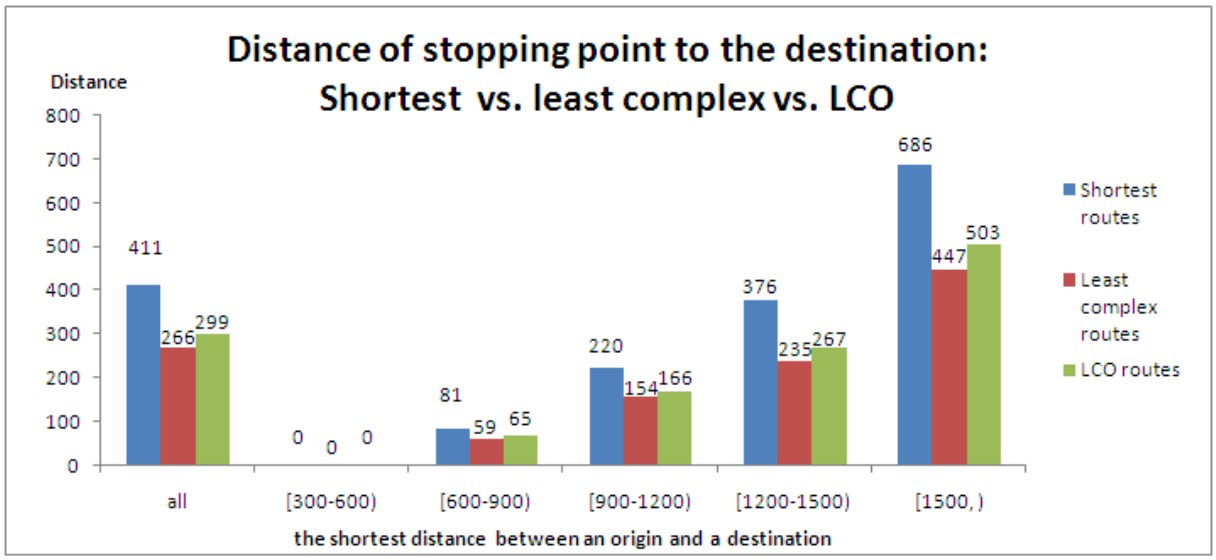

Figure 15. Distance (in metres) of stopping point to destination, comparing shortest routes, least complex routes and LCO routes.

In short, for successful travels, navigators using the LCO routes travel less distance compared to navigators using the other two kinds of routes. For metrics of total distance travelled and distance of stopping point to the destination, the difference between the least complex route and the LCO route is considerably smaller. However, for all the metrics, using the LCO routes leads to a better performance compared to using the shortest routes.

4.4 Summary

In summary, the main findings of the simulations are as follows:

- Compared to the shortest routes, the least complex routes and the length-complexity-optimized (LCO) routes lead to less complexity rating, and thus increase the chances of reaching the destinations when traversing the routes (Figures 5, 6, 7, 12 and 13).
- Compared to navigators using the shortest routes, when successfully reaching the destinations, navigators using the least complex routes and the LCO routes make fewer mistakes and travel shorter distance (Figures 8 and 14).

- Compared to navigators using the shortest routes, when failing to reach the destinations, navigators using the least complex routes and the LCO routes get closer to the destinations (Figures 9 and 15)[4].
- Compared to the least complex route and the shortest route, the LCO routes can achieve a better trade-off between complexity and distance (Figure 12).

In conclusion, the least complex route and the LCO route lead to less complexity rating, and thus improve the performance of navigators' tasks (more chances of reaching the destination, fewer mistakes made and shorter distance travelled).

Some drawbacks of the current simulations have to be noted. Literature has shown that the perceived complexity of a DP is often affected by many factors, such as signage (e.g. street names), spatial layout (e.g. the number of outgoing branches at the DP), visibility, differentiation, and whether there are landmarks at the DP (Montello 2005, Golledge 1999, Raubal and Winter 2002). In this paper, a navigator' complexity rating for a DP was simulated simply by considering the number of outgoing alternatives at the DP. Another criticism for the above evaluation is the simple behaviour designed to simulate human route-following. In the real world, humans often employ some other strategies when following a route (Montello 2005, Klippel et al. 2009). However, to some extent, the simulated complexity ratings partly reflect human's judge on complexity of DPs, and the simulated navigator also imitates some simple behaviour of human route-following. As the goal of the paper is to show how UGCs can be used to assist pedestrians' wayfinding, we argue that the designed simulations are sufficient for evaluating the proposed methods. In the meantime, we also expect that, compared the simulations, evaluation in field with human navigators will provide stronger and more compelling results about the advantages of collective intelligence based routes. The main reason is that compared to simulated navigators, human navigators can give *more accurate* ratings to indicate their actual perceived complexity of a DP. In addition, human navigators will use *more advanced strategies* when following a route. We will investigate these aspects in a follow-up project.

[4] Compared to navigators using the least complex routes and the LCO routes, navigators using the shortest routes make more errors during navigation, and therefore they are more often needed to recover from errors, which really slows down their wayfinding progress.

5. Conclusions and future work

The ubiquity of mobile devices (such as cell phones and PDAs) has led to the introduction of LBS. In this paper, we focused on one of the most important LBS applications - mobile pedestrian navigation systems. Current mobile navigation systems often ignore the social navigation aspect (i.e. using other people's experiences) which however is often used in our daily life. In addition, in the era of Web 2.0, more and more UGC is created/generated in many LBS applications. Making use of these UGCs is becoming more and more crucial.

In recognizing these challenges, we proposed the collective intelligence based route recommendation methods. The methods can make use of UGC (reflecting other navigators' wayfinding experiences), and provide navigators with the least complex route and the length-complexity-optimized (LCO) route. It is important to note that the proposed methods can be applied to both outdoor and indoor pedestrian navigation.

In order to analyse and evaluate the proposed routes and human performance when following these routes, some simulated experiments using the street network of the first district of Vienna (Austria) were designed. The simulated navigator (agent) was implemented to imitate human navigators' behaviours. Therefore, the designed simulations are sufficient for evaluating the proposed methods.

Our expectation was that the collective intelligence based routes (i.e. the least complex route and the LCO route) lead to less complexity rating, and thus improve the performance of navigators' actual wayfinding process. The results of the simulated experiments confirmed our expectation. It showed that compared to the shortest route, the collective intelligence based routes have a significant improvement of the route quality (with less complexity rating), thereby more effectively supporting users' wayfinding tasks (more chances of reaching the destination, fewer mistakes made and shorter distance travelled). More simulations with different road networks and different behaviour need to be done. We also plan to test the methods in a real world setting with human navigators in the follow-up project named EmoMap.

The proposed methods can be further improved by the following aspects:

- **Considering implicit ratings**: The ratings on DPs can be contributed explicitly and implicitly. Explicit contribution requires users to provide ratings actively which brings some burden to users. For implicit contribution, the system tracks users' implicit feedback (e.g. moving tracks) to unobtrusively infer their ratings (Ovaska and Leino 2008). With more ratings available, the proposed methods can provide more appropriate routes to the current navigator.

- **Incorporating collaborative filtering**: The collective intelligence based route recommendation algorithms make popularity-based recommendation to individual users. They are not especially made for any particular user but all get the same recommendations (Ovaska and Leino 2008). To make more relevant recommendations, personalized collaborative filtering (CF) should be introduced. A vision for this is "other people similar to you often chose this route". CF includes two steps: 1) find out similar users (this step can be viewed as assigning the current user to a group), 2) carry out the "popularity-based recommendation" on this group of users. Therefore, the proposed methods can also be used in the second step of CF.
- **Context-aware route recommendation**: Context-awareness is a key requirement for LBS applications. For different contexts (e.g. weather, companion, and season), pedestrians may need different kinds of routes. When incorporating context-awareness into the proposed methods, two key issues have to be considered: context modelling (identifying relevant context parameters) and context similarity. With these, the proposed methods can be easily adapted for context-aware route recommendation.

Acknowledgements

This work has been supported by the UCPNavi project (funded by Austrian FWF), which investigates the problem of indoor navigation in a smart environment. We also thank Professor Jonathan Raper and three anonymous reviewers for their constructive comments.

Reference

Bilandzic, M. and Foth, M., 2008. Social navigation and local folksonomies: Technical and design considerations for a mobile information system. *In:* H. Stylianos and W. Steven, ed. *Handbook of Research on Social Software and Developing Community Ontologies*. IGI Global.

Budhathoki, N., Nedovic-Budic, Z. And Bruce, B., 2010. An interdisciplinary frame for understanding volunteered geographic information. *Geomatica*, 64(1), 11-26.

Burrell, J. and Gay, G., 2002. E-graffiti: evaluating real-world use of a context-aware system. *Interacting with Computers*, 14(4), 301-312.

Burrell, J., Gay, G., Kubo, K. and Farina, N., 2002. Context-aware computing: A test case. *In:* G. Borriello and L.E. Holmquist, ed. *UbiComp 2002*. Springer Berlin / Heidelberg, 1-15.

Dijkstra, E, 1959. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 269–271.

Dourish, P. and Chalmers, M., 1994. Running out of space: Models of information navigation. Short paper presented at *HCI'94*, Glasgow, UK.

Duckham, M. and Kulik, L., 2003. "Simplest" paths: Automated route selection for navigation. *In:* W. Kuhn, M. Worboys, S. Timpf, ed. *COSIT 2003*. Springer Berlin / Heidelberg, 169–185.

Espinoza, F., Persson, P., Sandin, A., Nystrom, H., Cacciatore, E. and Bylund, M., 2001. GeoNotes: Social and navigational aspects of location-based information systems. *In:* G. Abowd et al., ed. *Ubicomp*. Springer, pp. 2-17.

Gams, E., Rehrl, K. and Kaschl, D., 2007. Using other people's trails for navigation assistance. In: *Proc. of the 5th Geographic Information Days*, Münster, Germany.

Golledge, R., 1995. Defining the criteria used in path selection. *Technical Report UCTC No. 78*, University of California Transportation Center.

Golledge, R., 1999. *Wayfinding behavior: Cognitive mapping and other spatial processes*. Maryland: The Johns Hopkins University Press.

Gonzalez, H., Han, J., Li, X., Myslinska, M. and Sondag, J., 2007. Adaptive Fastest Path Computation on a Road Network: A Traffic Mining Approach. In: *Proc. of VLDB 2007*.

Goodchild, M., 2007. Citizens as sensors: the world of volunteered geography. *GeoJournal*, 69(4), 211-221.

Hansen, F., Bouvin, N., Christensen, B., Groenboek, K., Pedersen, T. and Gagach, J., 2004. Integrating the Web and the world: Contextual trails on the move. *In: Proceedings of the Hypertext 2004*. ACM Press.

Haque, S., Kulik, L. and Klippel, A., 2007. Algorithms for reliable navigation and wayfinding. *In:* T. Barkowsky et al., ed. *Spatial Cognition V*. Springer, 308–326.

Huang, H. and Gartner, G., 2009. Using Activity Theory to identify relevant context parameters. *In:* G. Gartner, K. Rehrl, ed. *Location Based Services and TeleCartography II – from Sensor Fusion to Context Models*, pp. 35–45. Berlin/Heidelberg, Springer LNG&C.

Höök, K., 2003. Social navigation: from the web to the mobile. In: G. Szwillus, J. Ziegler, ed. *Mensch & Computer 2003: Interaktion in Bewegung*, 17-20.

Klippel, A., Hansen, S., Richter, K. and Winter, S., 2009. Urban granularities: A data structure for cognitively ergonomic route directions. *GeoInformatica*, 13(2), pp. 223-247.

Kounadi, O., 2009. *Assessing the quality of OpenStreetMap data*. Master thesis, University College of London.

Letchner, J., Krumm, J. and Horvitz, E., 2006. Trip Router with Individualized Preferences (TRIP): Incorporating Personalization into Route Planning. In: *IAAI'06*, AAAI Press.

Raubal, M. and Winter, S., 2002. Enriching Wayfinding Instructions with Local Landmarks. In: M. Egenhofer, D. Mark, ed. *GIScience*, Springer, pp. 243-259.

Montello, D.R., 2005. Navigation. In: P. Shah, A. Miyake, ed. *The Cambridge handbook of visuospatial thinking*, Cambridge University Press, pp. 257-294.

Ovaska, S. and Leino, J., 2008. *A Survey on Web 2.0*. http://www.cs.uta.fi/reports/dsarja/D-2008-5.pdf, Accessed on 12.2010.

Poser, K. and Dransch, K., 2010. Volunteered geographic information for disaster management with application to rapid flood damage estimation. *Geomatica*, 64(1), 89-98.

Raper, J., Gartner, G., Karimi, H. and Rizos, C., 2007. Applications of location-based services: a selected review. *Journal of Location Based Services*, 1(2), 89-111.

Surowiecki, J., 2004. *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations Little*. Brown ISBN 0-316-86173-1

Svensson, M., Höök, K., and Cöster, R., 2005. Designing and evaluating Kalas: A social navigation system for food recipes. *ACM Transactions on Computer-Human Interaction*, 12(3), 374-400.

Wexelblat, A., 1999. *Footprints: Interaction history for digital objects*. Thesis (PhD). Massachusetts Institute of Technology.

Wikipedia, 2011. *Dijkstra's algorithm* [online]. Available from: http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm , Accessed on 08.2011.

Winter, S., 2002. Modeling costs of turns in route planning. *GeoInformatica*, 6(4), 345-361.

Yuan, J., Zheng, Y, Zhang, C., Xie, W., Xie, X., Sun, G. and Huang Y., 2010. T-Drive: Driving Directions Based on Taxi Trajectories. In: *Proc. of ACM SIGSPATIAL 2010*.

Ziebart, B., Maas, A., Dey, A. and Bagnel, J., 2008. Navigate Like a Cabbie: Probabilistic Reasoning from. Observed Context-Aware Behavior. In: *Proc. of Ubicomp 2008*.