Elevation Enabled Bicycle Router Supporting User-Profiles

Nikolaus Krismer
Department of Computer Science, University of Innsbruck, Austria
nikolaus.krismer@uibk.ac.at

Doris Silbernagl
Department of Computer Science, University of Innsbruck, Austria
doris.silbernagl@uibk.ac.at

Martin Malfertheiner
Department of Computer Science, University of Innsbruck, Austria
martin.malfertheiner@student.uibk.ac.at

Günther Specht
Department of Computer Science, University of Innsbruck, Austria
guenther.specht@uibk.ac.at

ABSTRACT

Routing engines offer various algorithms to find the shortest or fastest path from point A to point B. Most of them are designed to find best paths for automobiles. Searching for a path as a cyclist has often very disappointing results. The main problem is that currently available routers do neither consider elevation or nor support profile aware routing. Therefore, this paper proposes a bicycle router that is build on top of GraphHopper, OpenStreetMap and SRTM supporting both of these requirements. The engine learns from previously gathered biking trips of users how fast one can ride on which street type as well as which kind of tracks are preferred. With the help of this information the system is able to suggest appropriate paths for each individual and estimates very accurate travel times.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Routing and layout*; H.2.8 [**Information Systems**]: Database Applications—*Spatial databases and GIS*

Keywords

OpenStreetMap, Routing, Algorithms, Elevation, User profiles

1. INTRODUCTION

People often rely on online services that propose best paths while planning a trip or when traveling. Contrary to traveling by car, cycling depends on different factors. One of these is elevation. Most cars have enough power to drive close to the speed limit regardless of the slope of the street. However, cycling uphill or downhill makes a huge difference and many of the currently available online routers do not consider an altitude profile at all.

Even though there are some elevation aware routing engines like GraphHopper[8], its estimations are not tailored to individual persons. The physical capabilities of Sunday cyclists, hobby bikers and professionals are very different and should be reflected by a routing engine. In addition, it should also be able to propose paths that mirror the preferences of the requesting cyclist.

Therefore, a bicycle routing engine is presented in this paper that addresses these issues. The system does not simply find the fastest route from point A to point B, it is able to find the best path for each user individually. The result is a combination of the users preferred street types, the accurate time estimation and the elevation profile. To achieve this goal as much information as possible provided by various datasets should be used.

The remainder of this paper is structured in five parts, with a related work section following this introduction. It includes a description of the routing engine GraphHopper and a short overview of other currently available research projects on profile aware routing. The third section explains the developed bicycle router in detail and gives insight into how profiles are stored and loaded during the route computation. The proposed bicycle router is later evaluated in Section 4 using various tracks and different bicycle riders. The paper ends with a short summary and possible future work to further improve the elevation aware routing engine.

2. RELATED WORK

Every routing engine needs a street network to find the shortest, fastest or most safe cycle path between two points. Therefore, the community driven geographic datasource from OpenStreetMap (OSM)[1] will be used. The project gets its data from individual supporters, but also from institutions. The work of Corcoran et.al.[3] shows that the community is very active and constantly increasing the level of detail. A typical OSM file contains lists of nodes, ways and relations, where each one of these elements can be further specified by a very flexible key-value pair tagging system. A node is a single point on the world, defined by latitude and longitude. A way is not only used as a way in the conventional sense, like a street, but it is also used to describe an area, like the

[1] `http://www.openstreetmap.org`

28^{th} GI-Workshop on Foundations of Databases (Grundlagen von Datenbanken), 24.05.2016 - 27.05.2016, Noerten-Hardenberg, Germany.

perimeter of a park or a forest. The third element used, the relation, is the most complex data element in OSM, as it can have nodes, ways and relations as members.

A lot of research about the quality of OSM has been done during the last years. Barron et al. [1] developed a tool to measure the quality of OSM regions based on the editing history. The authors pointed out that for each type of application, different requirements for the data hold. Completeness, currentness and logical consistency of the road network, attribute and positional accuracy are the major requirements for a routing engine. Loidl et al. [11] performed an evaluation on incomplete, erroneous and heterogeneous attributes and were able to achieve improvements for the investigates region. Since OSM is the best free available solution available it will be the base geographic source.

Every OSM node usually has a value for latitude and longitude, but although generally supported by OSM, only 0.07 % contain information about the third dimension: the elevation. Without accurate elevation profiles the system can not calculate correct time estimations. Thus, another data source that was collected in 2000 during the Shuttle Radar Topography Mission (SRTM) is used. Together, the two datasets OSM and SRTM, deliver enough information to create the required three dimensional geographic map on which the routing engine can search for paths. To increase the accuracy of the elevation model some post–processing has been performed that is discussed in detail by [17].

OpenStreetMap is used by many open source routing engines, such as OSRM[2], MapQuest[3], GraphHopper, BRouter[4] and many more. Most routing engines share a vast amount of similar characteristics, but often their focus is on cars and less on bicycle routing[13]. The elevation profile of a highway should not be neglected, but it influences the drive by car less than a cycling trip.

GraphHopper is chosen over the other available options as base engine because it is easily extendable, well tested and can deal with altitude profiles (in a very basic way) without modification. Besides, it has an active community that is bigger than the ones of the other engines capable of providing routes for cyclists. It uses a PBF (Protocolbuffer Binary Format) file of the area the user wants to search for routes in, combined with SRTM or CGIAR (Consultative Group on International Agricultural Research) elevation data. On startup of the routing engine, two initialization phases are performed. While marking nodes as tower or pillar nodes and maintaining tag restrictions in the first phase, a routing graph is built using the gathered information in the second phase. Weighting is then used to update the speed property of edges. Using the GraphHopper engine in its original and unmodified version, only the altitude of the starting and ending point is taken into account to calculate the slope of an edge. Depending on the result the speed is increased or decreased. For routing purposes several algorithms for finding routes are supported. The default algorithm used is bidirectional Dijkstra[4], but GraphHopper also supports unidirectional Dijkstra, one-to-many Dijkstra, as well as uni– and bidirectional AStar[6]. The algorithms calculate a weighting value for each path and the one with the lowest value will be returned to the user.

In the last few years there has been a strong movement on cycle route planning depending on security issues along the route. Researchers gathered information through surveys, cycling organizations [18] or in corporation with governments [10] [9] and built models for safe street networks. Routers were developed that are able to provide tracks with low traffic, routes labeled for bikes, broad paths and other metrics, but they never consider the individual preferences of a user. Some projects investigate route choice behavior of cyclists to support governments on taking street network modeling decisions. Hood et al. [7] did an analysis in San Francisco using GPS data of hundreds of participants. Community driven route planning is another approach to find better suited paths for cyclists. It uses a mobile application to record GPS data, geo-tagged media, noise level and roughness of a bike trip. A user then can see all tracks used by anyone beforehand and is able to lookup the information collected [15]. Another approach by the community is Cyclopath, a so called GeoWiki. A user can add information about points of interest on a track, post comments (width, surface) and rate the "bikeability" of the track [14]. All this user input can then be considered during route finding, but it again is not tailored to a specific user.

3. THE BICYCLE ROUTER

An elevation aware router for bicycles needs detailed information about the elevation profile and the properties of a street. On the one hand the router should avoid unnecessary climbings, but on the other hand a cyclist does not want to take too much of a detour. Even more important is the type of the street. Broach et. al. [2] collected rider preferences and showed that distance, turn frequency, slope, intersection control and traffic volumes have a strong impact on route choice. A router that is able to consider all these aspects, needs to be fed with data from OpenStreetMap as well as with elevation information.

One characteristic of GraphHopper is that it is designed to route with precalculated values for speed, distance and priority. These values are sufficient to find the fastest and shortest paths without knowledge of user preferences and cycling capabilities, but to find routes tailored to specific needs more data needs to be taken into account. Therefore, many tags of OSM were investigated regarding route finding. Table 1 lists the keys of the tags chosen and gives a short description for each one.

However, it is not sufficient to solely rely on information from these tags. Although a racing bike requires paved surfaces and mountain bikes can ride on paved as well as on unpaved surfaces, this is only one aspect of the criteria to consider. When it comes to profile based routing, more conditions need to be addressed. An easy path might be doable for a non-professional mountain biker and could be a nice alternative to a parallel, traffic loaded road, but a downhill racer might search for different way types. Therefore, it is obvious that additional and more detailed information for each edge (besides speed and its length) need to be stored.

3.1 Added routing information

An edge can not simply be extended with a ton of new information from various OSM tags. The added information has to be reduced to a minimum in order to hold the data in main memory. This allows the engine to operate in an efficient way. Therefore, a newly created flag encoder stores

[2] `http://project-osrm.org/`
[3] `http://www.mapquest.com/`
[4] `http://brouter.de/`

Table 1: Relevant OSM Tags

Key	Description
highway	defines the street network. Possible values are: motorway, trunk, primary, secondary, tertiary, unclassified, residential, service, living_street, pedestrian, track, road, footway, steps, path, cycleway
tracktype	is frequently used in combination with minor roads. It measures how well-maintained a way is (value ranges from one (very good) to five (very bad))
surface	can have a general value like paved or unpaved, but can also carry detailed information like asphalt, metal, grass, sand, etc.
smoothness	evaluates the physical condition of a streets surface with respect to a wheeled vehicle. Values range from excellent to impassable
sac_scale	is important regarding the difficulty of a path. Possible values are hiking, (demanding_)mountain_hiking, (demanding/difficult)_alpine_hiking
mtb:scale	is used in combination with track and path to specify the difficulty of a track in terms of mountain-biking. The value goes from zero (easy) to six (impossible for a classic mountain bike).
network	marks different cycle networks: icn (international), ncn (national), rcn (regional), lcn (local)

the way type, the adjusted speed depending on the ways surface, average incline/decline elevation and the distance an edge continues to rise.

3.1.1 Way Type

It is possible to meet the users preferences without storing the entire flood of OSM information. Dealing with bicycles it is very important to know the compactness of the way as well as the traffic load and the difficulty of a street (e.g. if a user is a passionate racing cyclist, then only those streets with paved surface should be considered). With that in mind 16 different classes have been designed. Each way gets assigned to one of these classes.

3.1.2 Surface

The speed calculation carried out to estimate travel times depends on way type and especially on its surface. A track with surface "compacted" and one with surface "mud" are for example both considered unpaved tracks, but riding them makes a big difference in terms of speed. A factor is defined for each surface/way type combination to modify the speed when riding it. Track surface "compacted" uses a multiplication factor of 1.2 while "mud" is assigned to 0.6, resulting in different calculation speed.

3.1.3 Elevation

The last information added for each edge is the elevation profile. Every point of an edge already has altitude information, but calculating the slope for each edge every time a request is processed would be a waste of processing power. The slope can be precalculated because it does not change between different users unlike way type preference.

The algorithm calculates the average incline and decline slope for each edge. Moreover, it stores how long an edge continues to rise. This is expressed as percentage of its distance. These three values are enough to reason about the edges elevation profile during path finding. Exceptions are made for tunnels, bridges, steps and very short edges (less than one meter). For these types of ways the slope will not be calculated, but instead considered to be flat.

The three elevation variables take up 19 bits of the flag variable on each edge, but deliver very valuable information for bicycle routing and avoid a lot of processing time.

With only 32 bits per edge (way type (4 bits) + speed (9 bits) + elevation (19 bits)) the router can reason on a very detailed street graph. The combination of all these variables allow path finding algorithm to retrieve the most suitable path for each user.

3.2 Routing without profiles

In order to profit from the newly generated details for each edge, the routing engine had to be extended with a new weighting implementation. Every path finding algorithm calls it to find the best path. Usually fastest routing takes the travel time for each edge as parameter while shortest path weighting considers only the distance of each edge. The developed approach also considers way type, surface and elevation.

Without any information about a user, the weighting algorithm prefers paved, low traffic, short travel time and bicycle designated streets. Therefore, a preference value depending on way type, surface type and elevation profile is calculated. Every edge starts with a priority value equal to four. The current implementation for non–profile based routing assigns a value between -4 and +3 to each way type. For example: the algorithm subtracts two from ways with unpaved surface and also penalizes if it has very steep inclines or a bad surface. The idea is that riding uphill and/or on unpaved surfaces is very inconvenient with a bicycle.

The resulting priority (a value between zero and seven that is divided by seven so it can be thought of as percentage value) is used to influence the travel time. This causes bicycle designated, low traffic, well paved, easy elevation and low travel time tracks to be preferred over other paths.

3.3 Profile based routing

Riding a bicycle is a very individual task. A hobby cyclist might take one hour for a track, a professional racer takes twenty minutes and a person new to cycling takes one and a half hours. Besides, a router should suggest different routes for a mountain bike and a racing bike. To support profile aware routing it is necessary to find a simple mechanism, that is powerful enough to create a detailed profile of a cyclist and can be used by the implemented weighting algorithm.

3.3.1 Profiles

Although OSM properties that are of interest are known, it would be too difficult for a user to define his preferences based on them. Setting the appropriate speed by hand would also be a cumbersome task for a person. Manually defining a profile is no choice for a user–friendly system so a way to create the profile automatically needed to be found.

Therefore, a strategy is developed that utilizes tracks which can be created using a tracking device such as a mobile phone or a GPS tracker. The big benefit of this approach is that the cyclist neither needs to specify any preferences on way types nor has to enter speed values for each way type and slope. Even better, the profile will evolve with every added track. Its the algorithms responsibility to extract the necessary information and model the users preferences and capabilities. Once a profile has been fed with enough information the router can propose customized routes with accurate travel time.

The basic profile, which is stored on disk, is a simple 16x61 matrix. Every row of the matrix corresponds to a way type and every column to a slope value ranging from -30% to +30%. A cell is either null (no data inserted so far) or contains distance and speed. Once a user uploads a track, the system extracts slope, speed (directly from the file provided) and way type (using GraphHoppers Map–Matching library) and fills the profile matrix with the calculated values.

3.3.2 Profile preparation

The profiles encoding the history of the cyclists have to be post–processed so that they can be used during the weighting process. Doing this, two major problems need to be addressed:

- The profile will probably have empty spots for certain way type/slope combinations even if the user uploaded a lot of different tracks.
- Faulty speed values need to be corrected, especially if only few samples have been uploaded for specific way type/slope pairs.

The solution to both of these problems is curve fitting. By specifying an accurate curve, gaps can be filled appropriately, outliers can be corrected and the speed value can be forced to adopt according to the well-known energetics of cycling that were explored by [12] and [16].

The researchers showed that cyclists gain speed quickly once the track goes downhill, but also that the absolute speed increase diminishes while the downward movement increased, because a cyclist starts to break and head wind becomes a factor. With increasing slope the average speed of a cyclist decreases fast but also - even in this case - the steeper the track gets the absolute speed decrease minimizes. This behavior is very similar to an inverted Sigmoid–curve [5]. In order to fit this S–shaped curve to the profile samples, a special version of the Sigmoid–curve, the so called inverse logistic function, is used:

$$f(x) = 1 - \frac{L}{1 + e^{-k(x-x0)}} \quad (1)$$

- L = the curves maximal value (the upper bound).
- k = the steepness of the curve (its "growth rate").
- $x0$ = the mid-point of the curve (its inflection point) .

Function 1 is used by the curve fitting process, searching for the parameters $L, k, x0$ using a weighted least squares method for every way type. Profile entries accumulated from multiple track parts are considered more important so a weighting factor ($distance\_per\_way\_type/speed$) is used. However, this method only works reliable if there is a certain amount of entries available and if those entries are well distributed regarding the ways slope. In order to prevent misleading fittings only those way types that have at least a total distance of ten kilometers will become part of the profile. Since this threshold alone does not guarantee a good fit (because the distribution of the entries itself has an impact too) artificial values created by GraphHoppers general speed function are added to the profile as well. These values are added with a short distance of fifty meters, so they become less important during the weighting process as soon as recorded tracks are added to the profile.

After this preparation the original profile with all the empty slots and faulty speed values is ready to be used by the weighting algorithm. The inverted Sigmoid–curve reflects the relation between slope and speed in terms of cycling. The threshold on traveled way type distance provides a reliable result.

3.3.3 Profile aware routing

The first step of GraphHoppers weighting mechanism asks the profile manager for the weighted average speed of the user at a certain way type and slope. If the requested way type has at least ten kilometers of uploaded track parts then the speed value will be returned from the curve fitted dataset. Otherwise the system searches for the way type with the highest amount of uploaded track parts. If this way type meets the required ten kilometers, the system takes the speed from this dataset and modifies it according to the difference between the base speed of the requested way type and the way type from the dataset. Only when dealing with a profile which does not include a single way type matching the ten kilometers requirement the routing engine uses the general (not user specific) speed calculation function. In any case the returned speed is used to calculate the travel time for this specific edge.

The second step estimates the preference of an edge. For this purpose the ratio of each way types distance to the total cycled distance of the user will be compute. This value influences the preference of a certain edge, but considering only the way type is not enough. The ratio of the surface type is used as well, which causes the routing engine to prefer also similar way types.

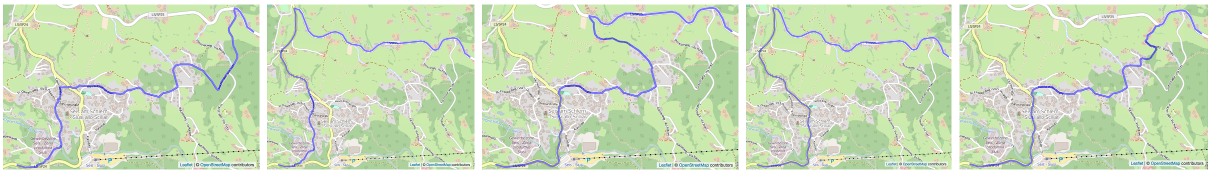
4. EVALUATION

The main purpose of the presented router is to provide user specific paths, with accurate travel time taking elevation into account. The estimated travel times are now evaluated against actually recorded track times. It is also examined how the router reacts according to different way type preferences of cyclists.

4.1 Travel time

Getting an accurate personalized travel time estimation is valuable to cyclists. In order to evaluate these estimations against real GPS recordings, ten trips have been recorded with different elevation profile and various difficulties. These recordings have been used to create three profiles.

- Profile 1 (P1) is created using two recordings. The



(a) orig. GraphHopper (b) Without profile (c) Moutainbike profile (d) Racing profile (e) Touring profile

Figure 1: Comparison between original GraphHopper and the bicycle router without and with profiles

first one is a long trip (about 15 kilometers and eighty minutes of travel time), with steep inclines as well as declines. The second track is shorter (approximately five kilometers) and almost flat.

- Profile 2 (P2) uses information from a track of approximately ten kilometers that has been recorded in both directions.
- Profile 3 (P3) consists of multiple short recordings, which have inclines, declines as well as flat parts.

Concerning the profile–awareness, four routes (which are not part of any of the profiles) have been requested comparing their results against actual recorded travel times. Table 2 lists the measured travel times and the ones for the created profiles. Moreover, it also contains the estimated time by the original, unmodified GraphHopper engine in contrast to the elevation aware router without profile as described in Section 3.2.

Table 2: Travel time (min) on recorded tracks

Track	GPS time	orig. Graph–Hopper	w/o profile	P1	P2	P3
Konstantin – Seis	12	20	16	13	17	13
Seis Dorf	14	26	18	14	16	15
Seis – Seiser Alm	71	130	111	67	70	66
Seiser Alm – Seis	15	43	18	14	15	15

The four different kinds of tracks in Table 2 cover uphill, downhill and flat paths, in and outside settlements. The third track is a steep uphill track, where routing with profile information makes a big difference. The bicycle router comes closer to the original travel time than the unmodified GraphHopper does for all tracks. This even holds true when the bicycle router without profiles. However, with only one exception (P2 on Track one is off by one more minute), profiles increase the accuracy of the estimated travel times even more.

To show that a profile makes a difference across cyclists two recordings of a very long trip from RideWithGPS[5] were used. Small parts have been extracted from these trips to become the test data, while the rest of the trips (excluding the test data) is used to create four profiles. The travel time estimation of the test data is then evaluated against the actual recorded time. The results are given in Table 3.

[5] `http://ridewithgps.com/`

Table 3: Travel time (min) with different riders

Experiment	GPS time	orig. Graph–Hopper	with profile
Rider #1 – uphill track	71	87	71
Rider #1 – downhill track	28	39	31
Rider #2 – up-/dowhill track 1	59	147	64
Rider #2 – up-/dowhill track 2	60	158	60

It can be observed that profiles work well for slow (Rider #1) and fast cyclists (Rider #2) and always deliver better results than the original, unmodified GraphHopper implementation.

4.2 Way type preference

One major goal for the presented bicycle router is that it should find the best path for each individual user. To evaluate this task, five profiles from different tracks have been created. They were used to look up routes using the same start- and end points. The resulting paths are analyzed to understand how the routing engine reacts to different user preferences. For this purpose the outcome of the unmodified GraphHopper, the developed bicycle router without a profile, with a profile of a racing cyclist and with a profile of a mountain biker have been compared in Figure 1. The original GraphHopper router (1(a)) proposes a path on streets that is very similar to the one of the bicycle router without profile (1(b)) and with a touring profile (1(e)). Figure 1(c) presents the suggested path using a MTB profile which is the only one, that proposes a track with unpaved surfaces. The last path to consider is the proposal for the racing cyclist (1(d)) that is a path on the primary roads in this area.

This shows that the routing engine reacts very flexible and responses according to a cyclists needs.

5. SUMMARY AND OUTLOOK

People rely on online services to plan their trips. A car routing engine does not really need to consider the individuality of a person, because travel time and route preference usually depends on predefined speed limits and real time traffic information. Most of the currently available routing engines take the same approach for bicycle routing, which results in very poorly estimated travel times and route proposals for different users. This paper showed how a profile aware routing engine can reflect the physical ability and the individual preferences of a cyclist. In order to find the best path for each person it is necessary to reason about the surface, the type of a way and the elevation profile, as well as the physical ability of the requesting cyclist.

OpenStreetMap has a very flexible data structure and in order to support user profiles it is necessary to include more than just the main classification of a street. The proposed approach can distinguish between paved and unpaved streets, high vs. low traffic streets, cycling ways and pushing sections. One of sixteen individual classes is assigned to each way. Combining this information with data about the ways surface as well as its elevation allows to find the most suitable route for racing cyclists, mountain bikers or any other category of cyclists.

With the help of user profiles the presented router is able to propose different routes according to the requesting person. The example from Section 4.2 shows that the routing engine reacts according to the preferred street types a cyclist drove on in the past. For the small search area (approx. five kilometers) it suggests five different tracks for five different profiles. The profile is also able to reflect the physical ability of a person, by collecting the average speed a user has on certain slopes and way types. Section 4.1 shows that it can estimate the travel time of a user with only a small error, which can further be neglected as a cyclist is not in the same condition every day or even due to other influences such as weather conditions. The results show that the system has a maximum error of five minutes for tracks lasting more than an hour. Without an accurate elevation profile it would not be possible to calculate such exact time estimations. Therefore, the elevation models have been examined in much more detail and were even improved in [17].

There are some minor issues that should be addressed in future works. Even though the router has the functionality to find the best path for a user, it is not yet suited to be used by the public, since profiles are created through command line operations and the user interface is not designed for daily use. The creation of a mobile application, that supports tracking, searching, uploading routes and managing the profile with a user-friendly interface could also be the next step to make the routing engine accessible to everybody. Another interesting idea of further improving path suggestions and the estimated travel time is to include real-time information, such as weather, traffic jams or the exclusion of closed roads. Also the fact that a cyclist can get exhausted over time and therefore is getting slower should be addressed by using an additional weighting factor related to traveling duration.

6. REFERENCES

[1] C. Barron, P. Neis, and A. Zipf. Comprehensive Framework for Intrinsic OpenStreetMap Quality Analysis. *Transactions in GIS*, 18(6):877–895, 2014.

[2] J. Broach, J. Dill, and J. Gliebe. Where do cyclists ride? a route choice model developed with revealed preference gps data. *Transportation Research Part A: Policy and Practice*, 46(10):1730 – 1740, 2012.

[3] P. Corcoran and P. Mooney. Characterising the metric and topological evolution of OpenStreetMap network representations. *The European Physical Journal Special Topics*, 215(1):109–122, 2013.

[4] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[5] J. Han and C. Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *From Natural to Artificial Neural Computation*, pages 195–201. Springer, 1995.

[6] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2):100–107, 1968.

[7] J. Hood, E. Sall, and B. Charlton. A GPS-based bicycle route choice model for San Francisco, California. *Transportation letters*, 3(1):63–75, 2011.

[8] P. Karich and S. Schröder. GraphHopper. `https://graphhopper.com`, 03 2016.

[9] M. Loidl and B. Zagel. Wie sicher ist sicher? - Innovatives Kostenmodell zur Ermittlung des Gefährdungspotenzials auf Radwegen. In *AGIT 2010*, pages 394–403. Wichmann Verlag, 2010.

[10] S. K. M. Loidl, B. Zagel and J. Reithofer. Radlkarte Salzburg - Das Radroutingportal für die Stadt Salzburg. In *AGIT*, pages 456–461. AGIT (2013), 2013.

[11] B. Z. u. G. P. Martin Loidl, Stefan Krampe. Aufbereitung von OpenStreetMap-Daten für GIS-Modellierungen und Analysen. In *Angewandte Geoinformatik 2014*. Angewandte Geoinformatik (2014), 2014.

[12] M. Nüschler. Leistungsfähigkeit auf dem Rad am Berg: Vergleich zwischen Marco Pantani und Hobbyfahrern. *Schweizerische Zeitschrift für "Sportmedizin und Sporttraumatologie"*, 49(2):79–81, 2001.

[13] OSM Wiki. Routing/online routers — OpenStreetMap Wiki. `http://wiki.openstreetmap.org/wiki/Routing/online_routers`, 2016. 03/02/2016.

[14] R. Priedhorsky and L. Terveen. The Computational Geowiki: What, Why, and How. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, pages 267–276, New York, NY, USA, 2008. ACM.

[15] Reddy, Sasank and Shilton, Katie and Denisov, Gleb and Cenizal, Christian and Estrin, Deborah and Srivastava, Mani. Biketastic: Sensing and Mapping for Better Biking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 1817–1820, New York, NY, USA, 2010. ACM.

[16] H. Schlichting and R. Nobbe. Untersuchungen zur Energetik des Fahrrads. *Technic-Didact*, 8:225–230, 1983.

[17] D. Silbernagl, N. Krismer, M. Malfertheiner, and G. Specht. Optimization of digital elevation models for routing. *Tagungsband zum 28. GI-Workshop über Grundlagen von Datenbanken (28th GI-Workshop on the Foundations of Databases)*, 2016.

[18] R. Turverey, D. Cheng, O. Blair, J. Roth, G. Lamp, and R. Cogill. Charlottesville bike route planner. In *Systems and Information Engineering Design Symposium (SIEDS), 2010 IEEE*, pages 68–72, April 2010.