ForestMaps: A Computational Model and Visualization for Forest Utilization

Hannah Bast, Jonas Sternisko, and Sabine Storandt

Department of Computer Science, University of Freiburg (Germany)
{bast,sternis,storandt}@informatik.uni-freiburg.de

Abstract. We seek to compute utilization information for public spaces, in particular forests: which parts are used by how many people. Our contribution is threefold. First, we present a sound model for computing this information from publicly available data such as road maps and population counts. Second, we present efficient algorithms for computing the desired utilization information according to this model. Third, we provide an experimental evaluation with respect to both efficiency and quality, as well as an interactive web application, that visualizes our result as a heat-map layer on top of OpenStreetMap data. The link to our web application can be found under `http://forestmaps.informatik.uni-freiburg.de`.

1 Introduction

Recreation is an important part of human life. Most people spent a significant fraction of their recreation time in public spaces such as forest, parks, or zoos. For the authorities of these public spaces, it is important to have utilization statistics about which parts of these spaces have been visited or are going to be visited by how many people.

Such utilization statistics are useful for a number of purposes. For example, for the prioritization of maintenance works. Or for selecting proper locations for new construction works (e.g. a look-out or an inn) or facilities (e.g. litter bins). Forests, in particular, are also used for purposes other than recreation, most notably for logging and preservation. Here past and projected visitor information helps to find a meaningful assignment of the various parts of the forest to the various purposes. Indeed, the original motivation for this paper was a request from the German forest authorities for computing such usage statistics for exactly the named reasons.

Our problem could be easily solved if we could track the movements of each visitor in the area of interest. But for a large number of visitors this is practically infeasible, and it would also be a major privacy issue. Instead, our approach is to come up with a *computational model* for how many people move through an area of interest on which paths. The input for this model should be publicly available data. Once computed, we visualize our usage statistics as a heat map, overlaid on top of a standard map.

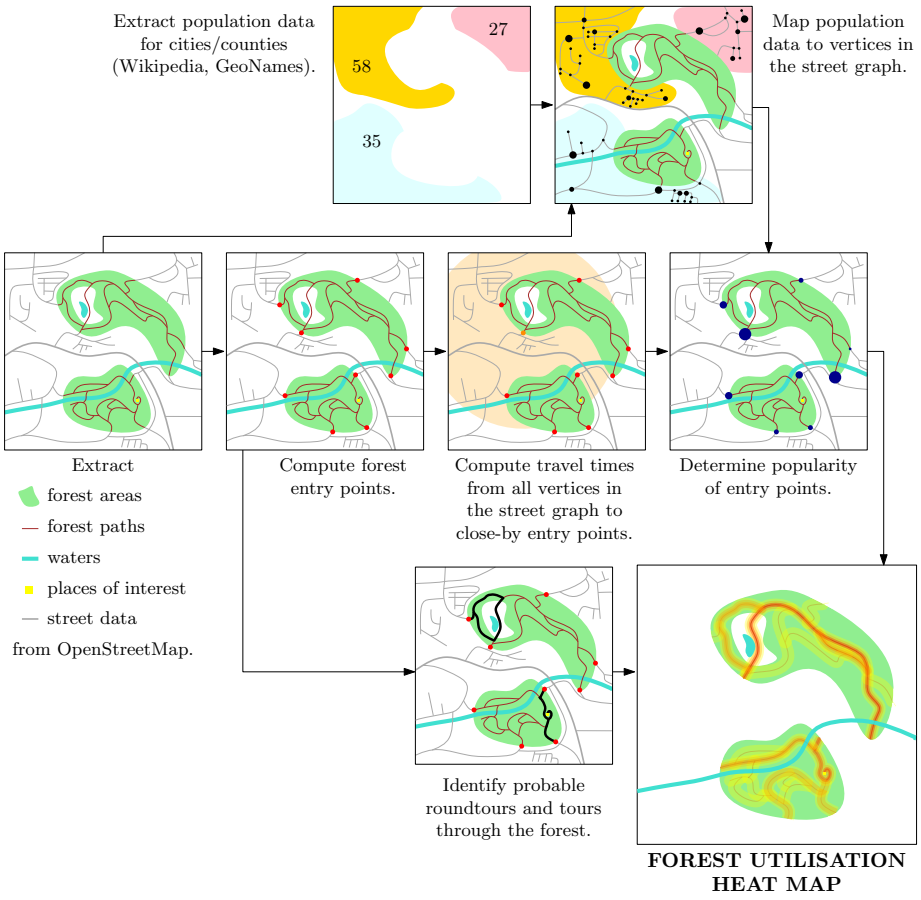


Fig. 1. Overview of the complete pipeline for our utilization distribution generator on the example of public forest areas.

1.1 Contribution

The contribution of this paper is threefold. First, we present a new model for utilization statistics of (paths in) public spaces based on publicly available data such as road maps and census data. We also show how to incorporate additional information like survey data (e.g. the mean time spent in the forest) or points of interests (e.g. look-outs or inns). Second, we developed and implemented a pipeline of efficient algorithms that uses this input data to compute the desired utilization statistics. Third, we provide an experimental evaluation on various data sets with respect to both efficiency and quality, as well as an interactive web application that is freely available. In particular, we created a web site which provides zoomable heat maps for all our data sets. The link, the details behind the heat map realization, as well as all our data sets and

code (thus enabling full reproducibility of our results) can be found here: `http://forestmaps.informatik.uni-freiburg.de`.

Throughout the paper, we will consider forest areas as a representative for public spaces. Forest areas are particularly hard to deal with, in particular harder than parks or zoos, for a number of reasons. First, forest areas have a large number of potential entry points, and it becomes part of our problem to determine these. Second, access to forest areas is usually unrestricted and there are no ticket booths or similar facilities, which could provide historical data on how many people entered at that particular entry point. Third, forest areas are often large, which entails a very large number of possible paths and round-tours. Since we want our algorithms to be efficient, we cannot simply enumerate these paths, but have to resort to more sophisticated solutions.

The mains steps of our pipeline are as follows. A schematic illustration is provided in Figure 1 above.

(1) Given a road map and the boundaries of the forest areas, compute the set of forest entry points efficiently.
(2) Given the population number of a whole area, compute a sound estimate of the distribution of inhabitants inside that area.
(3) For each forest entry point, use the road map and the result of (2) to estimate the number of people that are likely to use that entry point.
(4) Extract a representative set of routes and round-tours within the forest areas and estimate their relative attractiveness.
(5) Combine the information from (3) and (4) to estimate which parts of the forest areas are utilized to which extent.
(6) Visualize the utilization information from (5) in an intuitive and interactive manner in a web application.

We describe each of these steps in details in the following. Steps (1)-(3) are explained in Section 2. In Section 3, we propose two approaches for route and round-tour extraction as required in step (4), and show how step (5) can be accomplished on that basis. In Section 4, we explain how additional information like points of interest and survey data can be incorporated. Although this information is not crucial for our pipeline, it can enrich the model if available. In Section 5, we provide the setup and results of our experimental evaluation on three data sets, namely the road maps and forest areas of Germany, Austria, and Switzerland. In our evaluation, we consider both efficiency and quality. For our largest data set, the whole of Germany, our pipeline can be completed in about two hours. To estimate the quality, we compare our utilization information with GPS traces extracted from OpenStreetMap.

1.2 Related Work

From an algorithmic point of view, we are facing two main challenges. (1) Mapping population data given for an area to individual locations inside that area. (2) Computing a set of meaningful paths in the forest and their attractiveness.

Challenge (1) has been addressed in [1]. Here, aerial photographs are used as a basis to detect buildings in an area, and to extract building features like a building's footprint and its height. Given these characteristics, together with a pre-defined classification of buildings, the number of residents per block is computed. This approach is refined further by additionally considering city maps that distinguish between industrial and residential areas. This leads to a sophisticated multi-step algorithm that achieves very high accuracy. For over 90% of all buildings, the number of inhabitants is estimated correctly within a tolerance of 1 person for houses and 5 persons for apartments. The drawback of this approach is that it requires very sophisticated input data (in particular, high-resolution aerial images), which is not available for many areas. Also, this input data is large and complex and very time-consuming to compute with. In comparison, our approach is much simpler and uses widely available data like road maps and census information for whole countries. Thus we cannot, of course, estimate the number of residents for individual buildings. But we achieve good accuracy for estimating population distribution within sub-areas, see Section 2.4. And this is all we need for our purpose here: more fine-grained information would not help us to compute better utilization statistics.

Concerning challenge (2), several approaches to determine "nice" routes inside a given area have been developed. In [2], the problem of finding good jogging routes is investigated. It is first shown that an exact version of the problem is NP-complete. Then several simple and fast heuristics are proposed, which return useful routes in practice. These heuristics take as input a road map, attractiveness estimates for sub-areas, and a desired route length. In [3], a similar problem is addressed, namely tour suggestions for outdoor activities. The input there is a maximal tour length together with a tolerance value. The algorithm is based on spatial filtering and the computation of concatenations of shortest paths.

In both [2] and [3], the goal is to find few good routes or round-tours, or even just a single one. In contrast, for our problem we need to compute a comprehensive set of meaningful tours, from which we can then estimate the desired utilization statistics of the whole area. Previous work that comes slightly closer to this task is the computation of alternative routes in street networks. In [4], for example, the *via-node* approach is introduced. Here, all shortest paths from a given start s to a given target t via a third node v are computed, for every node v in the graph. Then a representative subset of paths is selected via criteria such as route length and spatial properties. We adapt the basic idea of this approach and turn it into a *via-edge* approach. We employ this approach to determine the degree of utilization of an edge inside the forest.

2 Computation of Entry Points and Their Popularity

In this section, we describe how to compute entry points and their popularity based solely on freely available data from OpenStreetMap. We already remarked above that for areas which require an entry fee, like amusement parks or zoos, this information about entry points is usually available. Entry points are then equal

to ticket booth positions and their popularity can be measured by the number of tickets sold. For freely accessible grounds like forests such data usually does not exist, and we need to estimate it by different means. To determine potential entry points, we compute the boundary polygon of an area and intersect it with the given path network. To determine the popularity of an entry point, we consider the population distribution in the surrounding area and the reachability of each entry point. Both of these are non-trivial procedures, and are described in more detail at the end of the section.

2.1 Extracting Street Networks and Forest Areas from OpenStreetMap

We evaluate our algorithms on data from OpenStreetMap (OSM). This project provides geographical information for nodes (in particular: latitude and longitude), polygonal paths (so called "ways", referencing previously defined nodes) and compositions thereof ("relations", referencing sets of nodes, ways or relations). The OSM data is provided in XML format. Each entity can have several attached tags, which are tuples of the form *key:value*. We parse all nodes, ways, and relations from the relevant OSM files and translate relations and ways to sequences of coordinates. We then build the road network from ways with a *highway:** tag. We generate the forest areas from entities with tags *landuse:forest* and *natural:wood*. Polygons with tag *boundary:administrative* are retrieved as boundaries of municipalities. Furthermore, we select nodes whose tags match certain combinations of tags as points of interest (POIs). For example, places with the tags *man_made:tower* and *tourism:viewpoint* are considered as POI. Note that our algorithms are independent of the particular data format and can also be applied when the data is given in other common GIS formats like ESRI shape files.

2.2 Computing Forest Entry Points (FEPs)

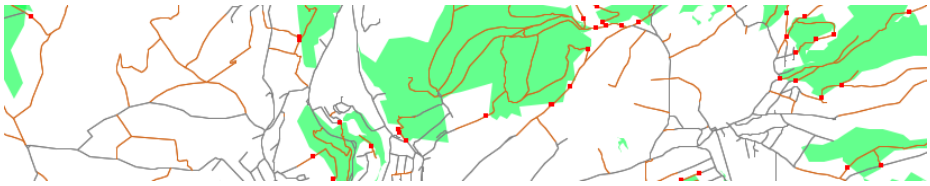


Fig. 2. Detail of a street network combined with forest areas (green). Red nodes highlight forest entry points. They are outside the forests, but have an adjacent edge that crosses a forest boundary.

The pre-processing of our input data, described above, provides us with the network of all paths, as well as polygons that bound the forest areas. We then find all edges from this network that intersect one of these polygons. The intersection

points are then our forest entry points (FEPs). Both the path network and the polygons consist of line segments. Our computation therefore reduces to computing the intersection between pairs of line segments. This can be done easily in constant time per pair. However, the naive approach of intersecting each path segment with each polygon segment would take "quadratic" time (number of path segments times number of polygon segments). This could be reduced by simple pruning techniques, for example, considering only path segments that lie in the bounding box of a forest area. But even then the computational effort would be too large. Also, forest areas extracted from OSM sometimes overlap, and the described pruning only works for disjoint areas. Identifying and merging overlapping forest areas first is again time-consuming.

We speed up this computation as follows. Instead of searching for intersections of line segments, we check for each node of the network if it falls inside a forest polygon or not. Using this classification, we iterate over all nodes in the graph and check for nodes outside the forest if they have an adjacent edge towards a node inside the forest. Such edges determine forest entry points. For the sake of simplicity, we do not add a new node for the crossing of the path with the forest boundary but instead use the last node on a path that is still outside of the forest. See Figure 2 for a classification example. To handle the large numbers of nodes for which the membership test has to be done, we rasterize the polygon to a bit array of sufficiently fine resolution (we used a precision of 10×10 meters per bit in our experiments). Consider this as image, where the forest areas are painted in white and the remaining parts in black. The membership test for a node is then reduced to a constant-time lookup of the value of the pixels at that node's coordinates. For our largest data set (Germany), we can thus compute all forest entry points in a fraction of the time needed for the whole pipeline (3 minutes of about 2 hours); see Table 3 in Section 5.

2.3 Incorporating Population Data

The next question is now many people are likely to use each of these FEPs. To answer that question, we need to know the population distribution in the surrounding area. Population numbers are widely available for larger administrative units (states, cities, villages), for example from the Wikipedia info boxes[1] or from GeoNames[2]. Using such data for our purposes entails two main challenges.

First, we need to know the boundary of the area to which a particular population pertains. Both kinds of data are available, but usually from different sources. Getting the proper assignment is non-trivial, because of different data schemas and variations in naming and spelling. We solve this problem by using an approximate match on a carefully selected set of name tags.

Second, we need more fine-grained information than just the population of a whole village. Specifically, we need an estimate of the number of people that live near each line segment from the given path network. We here make the following assumption, which we later validate in our experiments in Section 5.3.

[1] `http://de.wikipedia.org/wiki/Saarland`
[2] `http://www.geonames.org/2842635/saarland.html`

Assumption 1 *The population number in an area is strongly correlated to the accumulated length of local streets in this area. In other words: the population density and the density of local streets coincide.*

The intuition behind this assumption is that every house is typically close to some street, while the length of a street provides a good upper bound on the number of houses there and the density of houses does not vary too much in typical residential areas.

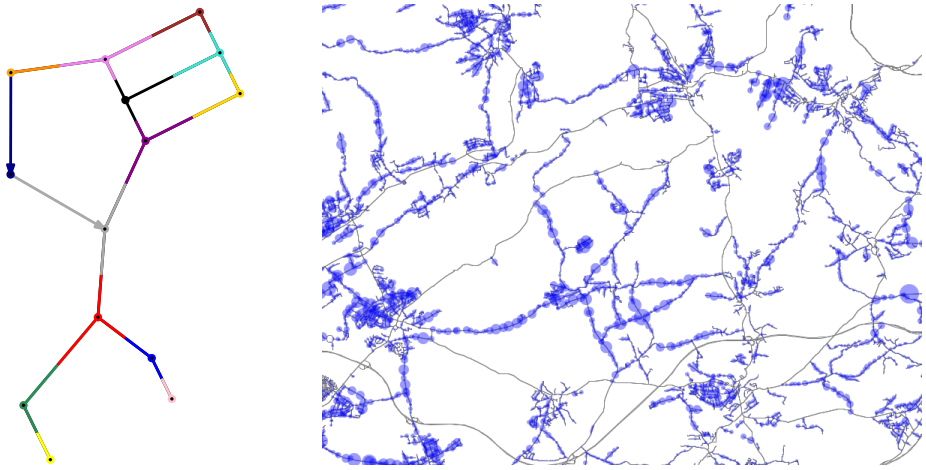


Fig. 3. Left: a small artificial example of a street Voronoi diagram. There is one distinct color for each vertex, and all the parts of edges belonging to the Voronoi cell of that vertex are drawn in that color. Right: a real-world population distribution based on such a street Voronoi diagram, where larger circles indicate higher population numbers.

We make use of this assumption as follows. For every street vertex, we compute the sum of the lengths of the street segments for which this vertex is the closest one. This is simply half of the sum of the lengths of all adjacent edges. From a geometric point of view, this amounts to computing a Voronoi diagram for all vertices and sum up the street lengths inside the respective Voronoi cells. For one-way segments we map the whole length to the tail node of that segment (since all residents must leave via that node). An example is given in Figure 3, left side. The running time of this approach is linear in the size of the vertices and edges in the street graph. It hence scales well to large networks. Dividing the computed sum of lengths for every vertex by the total sum of all edge lengths, we obtain percentage values. Multiplying these with the total population of the whole area results in an individual population estimation for each vertex. This is illustrated in Figure 3, right side. Types of streets, which are normally unpopulated, such as motorways, are simply excluded from the described procedure.

With the procedure as described so far, there is still some imbalance due to different building density along streets in different areas. For example, multi-story buildings with a large number of people are more likely in a city center, whereas houses in remote areas tend to be more sparse and to have less inhabitants. In an extreme case, like an industrial zone, there might be streets but no actual inhabitants at all. We alleviate this imbalance by identifying (large) clusters with a high density of living streets, typically metropolitan areas, using a simple grid-based approach. Inside of such clusters, we increase the percentage values by multiplying the above-mentioned lengths of segment sums by a constant *weight factor*, specified below. To identify such clusters, we again use a grid-based approach. We choose 1000×1000 cells. We say that a grid cell is *dense*, if the sum of the lengths of the contained streets is 25% or more above the average (taken over all grid cells that contain at least one street). We use a weight factor of 3 for such grid cells. We found this to be a typical ratio when comparing population count divided by total street length in city vs. rural areas. Moreover, we say that a grid cell is *super dense*, if the sum of the lengths of the contained streets is 50% or more above the average. We use a weight factor of 6 for such grid cells. This simple approach requires only constant time per edge and grid cell. In combination with the street Voronoi diagram, this gives us a very efficient tool for population estimation.

2.4 Computing Popularity Values for Entry Points

The closer someone lives to a certain FEP, the higher the probability that this person will use this FEP for visiting the forest. If several FEPs are nearby, the likelihood for usage is distributed among them. To compute for every street vertex $v \in V$ the set of suitable FEPs, we could execute Dijkstra's algorithm from each such vertex. For a more realistic model, we restrict the travel time to 30 minutes. For each FEP f contained in the Dijkstra search tree, we then compute the usage probability as

$$u(v, f) = 1 - \frac{d(v, f)}{\sum_{f' \,:\, d(v,f') < 30min} d(v, f')} \ ,$$

where d denotes the travel time between the given node and FEP. The popularity value of each FEP f is then computed as $\sum_v pop(v) \cdot u(v, f)$, where $pop(v)$ is the population for v computed as described in the previous section.

But as stated above, this straightforward approach to computing these popularity values would be to run a Dijkstra from each street vertex v. Even with a bounded radius of 30 minutes, the computation time would be several days for a data set like Germany. What comes to our rescue here is that the number of FEPs is about two orders of magnitude smaller than the number of all nodes in the street network. We therefore run a Dijkstra computation (bounded to a radius of 30 minutes) on the backward graph from each FEP. That way, each street vertex can (and will) occur in a number of Dijkstra results, typically on the order of hundreds. Explicitly storing all the values that contribute to the

$u(v, f)$ from above would hence consume many times more space than needed for storing the actual network.

To avoid this, we discretize the $d(v, f)$ distances into buckets with a resolution of 5 minutes. That way, we need to store only a few counts for each vertex. Namely, the number of FEPs reachable in less than 5 minutes, the number of FEPs reachable between 5 and 10 minutes, and so on.

It then remains to distribute the populations $pop(v)$ over the FEPs. This is easily done with another backwards Dijkstra from each FEP.

Figure 4 illustrates the backwards approach by a small example.

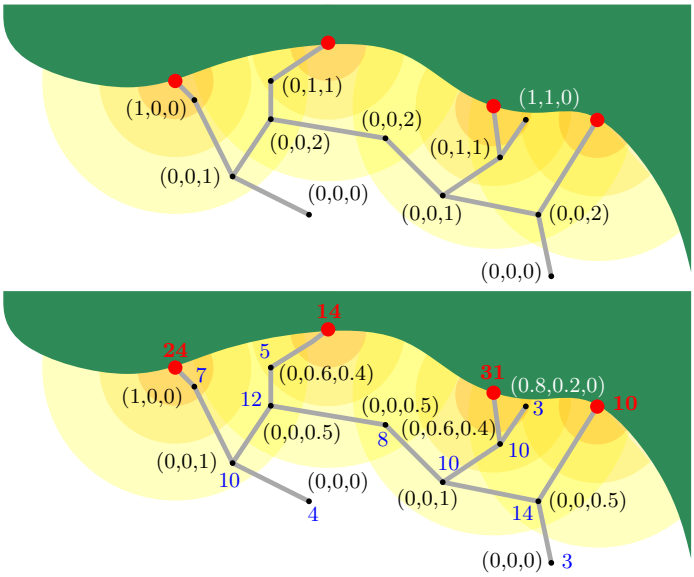


Fig. 4. Backwards approach for four forest entry points (red) and three travel time buckets (indicated by the circular areas with the color ranging from orange to yellow). The upper image shows the result of the first round of backward Dijkstras. For each street vertex (black), we obtain the number of FEPs in each bucket. The lower image shows the result of the second rounds of backward Dijkstras. The counters from above are converted to usage likelihood values. See for example the resulting tuple $(0, 0.6, 0.4)$ generated from the counters $(0, 1, 1)$. Here, we have one FEP reachable between 5 and 10 minutes, and another one between 10 and 15 minutes. Summing up the average travel time for these buckets, we get $7.5 + 12.5 = 20$ minutes. The likelihood for the FEP in the 5-10 bucket is then $1 - (7.5/20) = 0.625 \approx 0.6$. The likelihood for the other FEP equals the remaining probability of $0.375 \approx 0.4$. These values are then used as coefficients to map population values (blue) to FEPs. The resulting values are the popularity counts for each FEP (red).

3 Computing the Utilization Distribution

The utilization distribution that we want to compute depends on two main factors: the popularity of entry points and the attractiveness of paths through the public space. We have shown how to compute the popularity of entry points in the previous section. In this section, we show how to compute a comprehensive set of paths and round-tours and their relative attractiveness. We consider two approaches: *flooding* (Section 3.1) and *via-edge* (Section 3.2). Both approaches allow to combine entry point popularity with tour attractiveness values in order to estimate the desired utilization distribution better.

3.1 The Flooding Approach

The goal of this section is to assign to each edge in the forest an attractiveness value that reflects how likely this edge is included in a tour. A naive approach is to run a Dijkstra search from every FEP considering only the forest subgraph of the street network. Then for every edge (v, w) explored by this search, its attractiveness gets increased by the fraction of the popularity of the FEP f and the distance from f to w. Think of this technique as "flooding" the entry point's popularity along shortest paths into the nearby forest. Consequently, a higher FEP popularity contributes to a higher attractiveness of the edges in its range. Conversely, the further away an edge is from the entry points to the forest, the lower is its attractiveness. This approach is simple and efficient. However, it has the disadvantage that it might leave some edges with an attractiveness of zero, because they are not part of any shortest path (Figure 5). But it is not uncommon that people walk along non-shortest paths during leisure activity. As a remedy, we could compute the k shortest paths to each inner vertex, either forbidding loops [5] or allowing them [6]. Another way would be to define several edge metrics (length, niceness, quietness, ...) if such information is available, and then search for optimal paths for several linear combinations of these metrics. Both of these approaches would generate multiple paths between a FEP and a node inside a forest, but still there is no guarantee that each edge of a path is considered at least once. Moreover, this simple approach only models round-tours reasonably, whereas hiking from one FEP to another is not included. We therefore propose, in the following subsection, an alternative approach that captures tours through the forest as well.

3.2 The Via-Edge Approach

In the via-edge approach, we iterate over the forest edges one by one to calculate their respective attractiveness. Specifically, for each edge (v, w), we run a backwards Dijkstra from v and a forwards Dijkstra from w. This provides us with a set of paths of the type $\text{FEP}_1 \to^* v \to w \to^* \text{FEP}_2$ (where, of course, $\text{FEP}_1 = \text{FEP}_2$ is possible). Now for every such path we increase the attractiveness value of the edge (v, w) by the minimum of the popularity values of $\text{FEP}_1, \text{FEP}_2$ multiplied with the shortest path distance between FEP_1 and FEP_2, divided

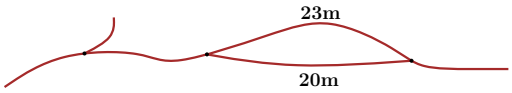


Fig. 5. The upper path is three meters longer than the lower one, so any Dijkstra-based point-to-point search will not include the upper path in a solution. Still, this path section might be used in a hike.

by the total path length (which can be extracted from the labels created in the two Dijkstra runs and the length of the edge). For round-tours ($\text{FEP}_1 = \text{FEP}_2$) the attractiveness increase would be zero, hence this case is handled like in the flooding approach. So the attractiveness increases with the popularity of the entry points, and the smaller the difference between the via-edge path and the shortest path the higher the attractiveness. That way, we consider all tours via a certain edge and we generate meaningful attractiveness values for all edges in the forest.

4 Incorporation of Additional Information

4.1 Survey Data on User Preferences

The two main behavioral factors that influence utilization intensities in forests are: first, the travel time to get to the forest, and second, the amount of time spend in the forest. If these two factors are captured by survey data, we can incorporate them into our pipeline and thereby increase the accuracy of our model. Table 1 shows the results of a survey on recreational behavior from a German forest authority[3] as published in [7]. We observe that most people prefer forest entries in the vicinity of their residences (about 80% take $\leq$ 15 minutes to get to the forest). The time spent in the woods has a peak between 30 minutes and one hour, and the distribution has a positive skew towards longer sojourns. We can plug in these observations as follows: for the computation of FEP popularity values, we use travel time buckets in the backwards approach as provided by the survey. After the backwards Dijkstra runs from the FEPs, we distribute the population values of each street vertex according to the percentage values given in Table 1 (if for a bucket no FEP is available, we add the percentage to the previous bucket). As an example, assume that from a vertex v we can reach one FEP f_A in less than 5 minutes and two FEPs f_B and f_C between 10 and 15 minutes. We conclude that 63% of the population of v use f_A, and 18.5% use f_B and f_C respectively. To incorporate the duration of stay times we use the percentage values of the second column of Table 1 to weight tours detected by our algorithms (*flooding* or *via-edge*). Thus, edges used in tours with a length favored by more people receive higher attractiveness values.

[3] `http://www.fva-bw.de/`

Table 1. Survey data about forest visits according to [7].

travel time to the forest		time spent in the forest	
up to 5 min	38%	up to 30 min	25%
6 to 10 min	25%	31 to 60 min	42%
11 to 15 min	16%	61 to 90 min	11%
16 to 30 min	18%	91 to 120 min	15%
longer than 30 min	3%	longer than 120 min	7%

4.2 Points of Interest (POIs)

Of course, the presence of attractors like an inn, look-outs or a lake in the forest, vivaria in the zoo or sunbathing lawns in parks, affects the likeliness of tours passing in the vicinity and thus the edge attractiveness. If such POIs are available, an easy way to incorporate them would be to assign a "niceness" value to each edge which increases with the proximity to an attractor. Then every tour created by our via-edge approach described above gets weighted with the respective niceness value of the edge. Unfortunately, this approach covers only tours visiting *a single* sight with our via-edge approach. But popular routes often pass multiple POIs, and this is not properly modeled by this approach. One remedy would be to compute the shortest paths via all permutations and subsets of attractors. But this would be too time-consuming as the number of possible tours grows exponentially in the number of POIs. So instead, we compute shortest paths between all pairs of attractors and increase the niceness of edges on these paths accordingly, which can be accomplished in polynomial time. In this way customer flows concentrates on paths between attractors and the utilization intensity increases on such sections.

5 Experimental Results

In this section, we evaluate our algorithms on real-world data with respect to both efficiency and quality. All our algorithms are implemented in C++, except for the raster-based FEP extraction described in Section 2.2, which is written in Java. As we see in Table 3 below, the running time of this component is only a small fraction of the total running time. All our experiments were performed on a single core of an Intel i5-3360M CPU with 2.80 GHz and 16 GB RAM.

5.1 Data

We extracted street networks and forest boundary polygons from OpenStreetMap (OSM) for three countries: Germany, Austria, and Switzerland.[4] Note that for all of these countries the available data for street networks and forest areas can be considered as almost complete. Surely, the magnitude of nodes, edges and

[4] Retrieved from `http://download.geofabrik.de` on November 26, 2013.

forest areas resembles reality. To get evidence for the scalability of our approach, we also performed some of our experiments on two states within Germany, one large (Baden-Württemberg) and one small (Saarland). Population statistics were looked up manually in Wikipedia. The main characteristics of our data sets are summarized in Table 2.

Table 2. Our five data sets (ordered by graph size) and their main characteristics.

Graph	OSM extract			Wikipedia extract
	nodes	edges	forest areas	population
Saarland	535,595	1,149,654	2,165	994,000
Switzerland	6,562,482	12,314,060	39,087	7,997,000
Baden-Württemberg	7,156,371	15,460,922	26,507	10,598,000
Austria	13,473,037	23,469,568	66,498	8,462,000
Germany	47,839,447	93,811,864	394,522	81,890,000

5.2 Running Times

We measured the running time for each step of our pipeline, excluding the data extraction from OSM. The resulting times along with the total running time can be found in Table 3.

Table 3. Running times for the main steps of our pipeline in seconds. The numbers for the "edge attractiveness" column are for our (more sophisticated) via-edge approach.

Graph	Runtime (in seconds)				
	population mapping	FEP extraction	FEP popularity	edge attractiveness	total
Saarland	0.01	1.92	63.53	40.65	2min
Switzerland	0.38	11.95	51.20	211.34	5min
Baden-Württemberg	0.43	20.78	584.65	1223.43	31min
Austria	0.79	24.15	56.78	912.55	17min
Germany	2.86	181.25	512.29	6958.33	135min

We observe that the fraction of time spent on the population mapping is negligible. It is just a few seconds even for our largest data set (Germany). This is easy to understand, since the computation of the (street) Voronoi diagram requires only a single sweep over all edges. The fraction of time spent on the FEP extraction is also relative small. This is due to the efficient rasterization approach described in Section 2.3. The resulting number of FEPs per input can be found in Table 4. This number influences the runtime of the subsequent steps, especially the popularity value computation which requires two runs of Dijkstra for each FEP. We observe that the number of FEPs for Austria, Switzerland and

Baden-Württemberg are very similar, despite their widely different number of forest areas. This is because the number of FEPs is influenced by both forest area size and street network density.

Table 4. Number of forest entry points for our test graphs.

Graph	number of forest entry points (FEPs)
Saarland	12,157
Switzerland	124,374
Baden-Württemberg	139,468
Austria	198,241
Germany	971,517

The two remaining steps, the computation of the FEP popularity values (Section 2.4) and of the forest path attractiveness values (Section 3), are the most time-consuming of our whole pipeline. This is because both steps require a large number of Dijkstra computations. Note that the numbers reported in Table 3 for the last step are for the more sophisticated via-edge approach, described in Section 3.2. The more simplistic flooding approach is about 40 times faster.

We also observe that the total running time for Baden-Württemberg (BW) is slightly larger than for Austria, despite the smaller street graph for BW. This is because the forest areas in BW are larger and more compact and with many more paths inside.

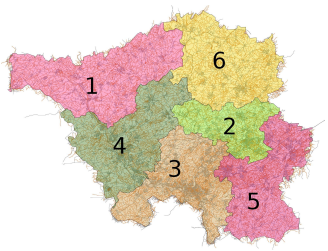
5.3 Quality of the Estimated Population Distribution

In Section 2.3, we described how to distribute the population given for a district over the individual nodes of the contained street network. We evaluated the quality of our approach as follows. We do not have access to the positions of individual buildings and the number of people inhabiting them. We instead manually researched the population for some lower administrative districts contained in our data sets. For each such district, we compared this number to the sum of populations we computed for all nodes of the contained street network. Table 5 provides the result of this comparison for the German state of Saarland and its six major sub-districts.

The average deviation is only about 10%, with a maximum deviation of 21% for one sub-district. This is surprisingly good, given our simple approach. We repeated the same experiment on the whole of Germany (81.9 million inhabitants) and chose 20 districts randomly with population ranging between 3.4 million ("Berlin") and 56,312 ("Wittmund"). On average, we over- or underestimated the population in a district by 28%, while never being away more than a factor of 2 from the correct result.

Table 5. Accuracy of our estimated population for the six sub-districts of the German federal state Saarland. Δ denotes the relative deviation of our result compared to the ground truth in percent.

Saarland	Population Number actual	estimated	Δ
1 Merzig-Wadern	103,520	96,903	−7%
2 Neunkirchen	134,099	118,628	−12%
3 Saarbrücken	326,638	315,866	−3%
4 Saarlouis	196,611	175,705	−11%
5 Saarpfalz-Kreis	144,291	157,316	+9%
6 St. Wendel	89,128	107,452	+21%



5.4 Quality of Our Estimated Utilization Distribution

Evaluating the quality of our final result (which parts of the given forest areas are used by how many people) is difficult, because there is no such empirical data available. Indeed, the lack of availability of such data is the main motivation for this work, see Section 1.

However, large numbers of GPS traces from people contributing to OpenStreetMap are publicly available[5]. Specifically, we used packages containing about 17,000 traces for Switzerland, 22,000 for Austria and more than 200,000 for Germany. Clearly these traces are susceptible to all kinds of bias and cannot be considered as a "ground truth". However, we found it quite safe to assume that highly frequented paths indeed correlate with prominent numbers of GPS traces. We hence proceed as follows. We provide two comparisons, one visual and one quantitative.

For the *visual* comparison, we produce comparable heat maps for the GPS data and the utilization intensities computed by our two approaches (flooding and via-edge). For the GPS data, we intersect the GPS traces with our extracted forest polygons and overlaid the traces, each with a low transparency. We then map the aggregated transparency values to the same color range used in the heat maps for our two approaches. For a fair comparison, the color values were normalized for each of the three maps individually. To avoid distortions by high intensities from a few outliers, the top 2% of intensities all get assigned to the most intense color from our color scheme (the reddest red in our pictures). The remaining intensities are mapped linearly to the remaining color range (from red over orange and yellow to no color).

Figure 6 shows these three heat maps for Baden-Württemberg. A number of interesting observations can be made.

(1) For both of our approaches, the hot spots are in similar locations as for the GPS data.

[5] `http://www.openstreetmap.org/traces` and `http://goo.gl/wczD8H`

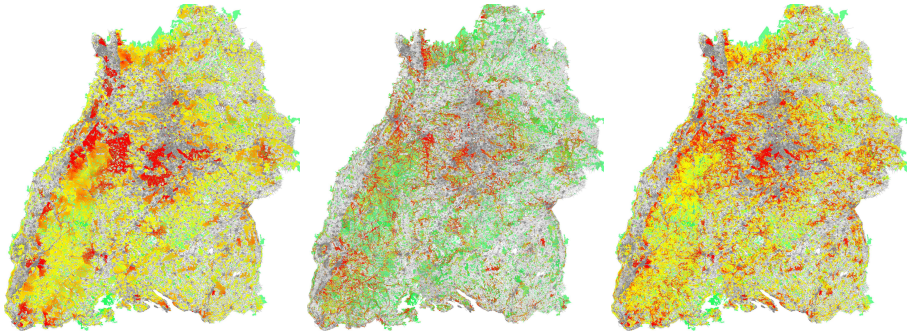


Fig. 6. Comparison of heat maps for the GPS data (middle), our simple flooding approach (left), and the more sophisticated via-edge approach (right). The intensities range from yellow over orange to red. The more red the more intense. Forest areas with no paths or no data are green. A detailed discussion is provided in the text below.

(2) The flooding approach tends to produce larger hot spots. This is an undesired artifact of the approach, arising from not considering tours through the forest, but only proximity of forest edges to FEPs.
(3) The via-edge approach tends to produce more pronounced hot spots, very similar to those from the GPS data. The via-edge approach also singles out specific paths and edges, differentiating attractiveness values much better than the flooding approach between several forest walks in the same area.
(4) Many of the smaller hot spots from the GPS data (e.g. those in the lower right part of the map) can be found in the heat map of the via-edge approach, but not in the heat map of the flooding approach. Especially in small forest areas, people tend to walk *through* the forest instead of making a round tour. Only the via-edge approach models this behavior satisfactorily.
(5) The coverage of the GPS data is very limited, the coverage of both of our approaches is very good.
(6) In the lower left part, parallel to the border, we see an intense forest usage indicated by the OSM traces, but only a small to medium intensity according to our models. This is at least partly a border effect, because the cut-off at the state boundary leads to smaller population values in the surrounding areas of forest entry points near this boundary.

Figure 7 shows an enlarged version of the heat map for Baden-Württemberg, as produced with the via-edge approach. Figures 8 and 9 show the respective heat maps for Austria and Switzerland. For Austria and Switzerland, we observe a large fraction of yellow areas, because there are far from populated areas. And again, border effects might play a role in some areas, as seen for Baden-Württemberg before. Furthermore, many forest areas in Austria and Switzerland are tourist spots. If corresponding tourist use data were available, it would make sense to include it into our model.

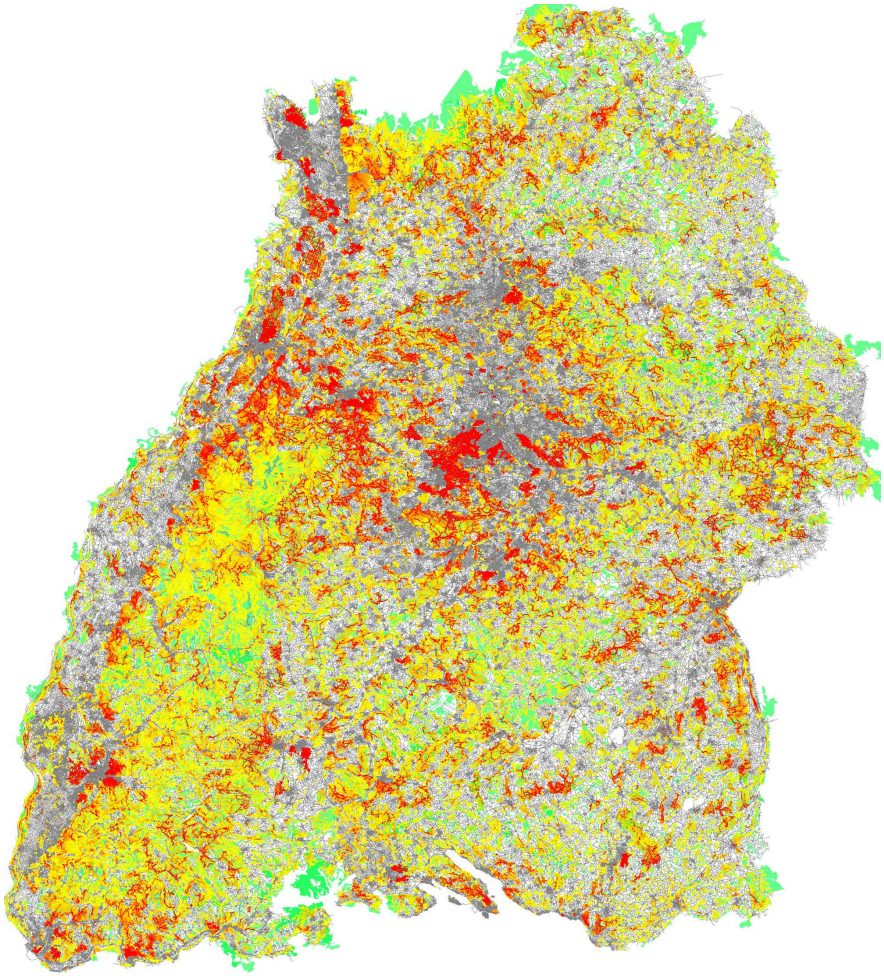


Fig. 7. Heat map for Baden-Württemberg, as produced with our via-edge approach.

For our *quantitative* comparison, we extracted the top-50 hot spots for the OSM trace map and for the heat maps computed by our two approaches. To extract the top hot spots, we laid a grid over the map and summed up the heat map intensities in each grid cell. For each of the three maps, we then extracted the 50 grid cells with the highest value. We then counted which percentage of the top 50 grid cells from the OSM trace map are also in the top 50 grid cells in our two approaches. The result is reported in Table 6. As in the visual comparison, also the quantitative results show a clear advantage of the via-edge approach over the simple flooding approach. However, this advantage is smaller than one might expect from the visual comparison, in particular Figure 6. This is because the

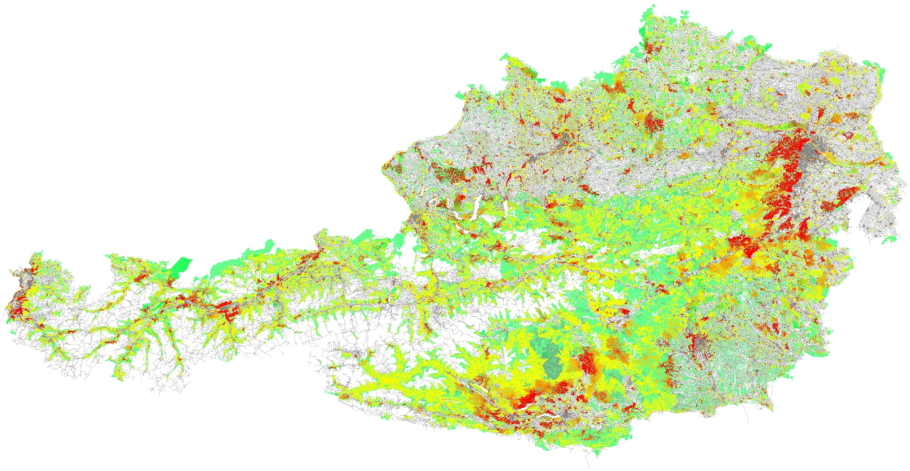


Fig. 8. Heat map for Austria, as produced with our via-edge approach.

grid discretization and the top-50 approach over-emphasizes the (easy) top hot spots and blurs the subtle differences between flooding and via-edge discussed above.

Table 6. Quality analysis of our two edge attractiveness models using a set of traces extracted from OSM as ground truth. Hot spot detection is evaluated against this baseline, the values are given in percent.

Graph	Edge Coverage			Hot Spot Detection	
	OSM	flooding	via-edge	flooding	via-edge
Saarland	37	99	100	27	38
Switzerland	28	97	100	20	24
Baden-Württemberg	29	95	100	22	31
Austria	22	93	100	18	23
Germany	26	91	100	17	24

6 Conclusions and Future Work

We have designed, implemented, and evaluated a pipeline of algorithms for estimating the utilization distribution of public spaces, in particular forests. We used only simple and publicly available map data as input. Our approach predicts not only the utilization distribution in an area but also the utilization on the fine-grained level of paths. We have also provided a visualization of our results in an interactive web application, along with all data sets and code needed for reproducibility. For future work, it would be most interesting to get hold of a

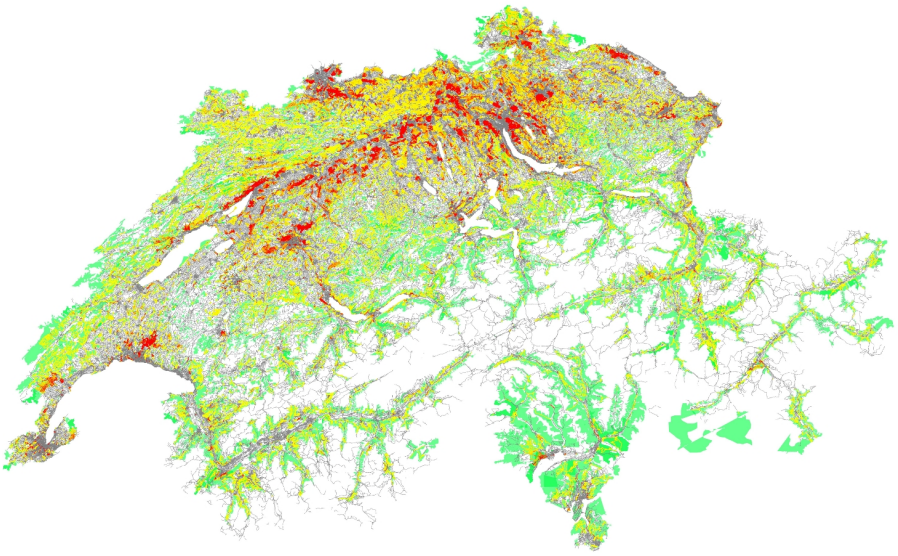


Fig. 9. Heat map for Switzerland, as produced with our via-edge approach.

more comprehensive ground truth (e.g. usage statistics from traces or polls), and use these for a more thorough quality comparison and to fine-tune our model and our algorithms.

References

1. Ural, S., Hussain, E., Shan, J.: Building population mapping with aerial imagery and GIS data. International Journal of Applied Earth Observation and Geoinformation **13**(6) (2011) 841–852
2. Gemsa, A., Pajor, T., Wagner, D., Zündorf, T.: Efficient Computation of Jogging Routes. In: Experimental Algorithms. Springer (2013) 272–283
3. Maervoet, J., Brackman, P., Verbeeck, K., De Causmaecker, P., Berghe, G.V.: Tour suggestion for outdoor activities. In: Web and Wireless Geographical Information Systems. Springer (2013) 54–63
4. Luxen, D., Schieferdecker, D.: Candidate sets for alternative routes in road networks. In: Experimental Algorithms. Springer (2012) 260–270
5. Yen, J.Y.: Finding the k shortest loopless paths in a network. management Science **17**(11) (1971) 712–716
6. Eppstein, D.: Finding the k shortest paths. SIAM Journal on computing **28**(2) (1998) 652–673
7. Ensinger, K., Wurster, M., Selter, A., Jenne, M., Bethmann, S., Botsch, K.: "Eintauchen in eine andere Welt" - Untersuchungen "uber Erholungskonzepte und Erholungsprozesse im Wald. German Journal of Forest Research **184**(3) (2013) 70–83