Enabling comparison of travel times for taxi and public transport

JOHANNA AXELSSON
Master thesis in computer science

Enabling comparison of travel times for taxi and public transport

Johanna Axelsson
joaxel@kth.se

Master’s thesis at Valtech AB
Supervisor: Jens Lagergren
Examiner: Anders Lansner

15th September 2015
Abstract
Taking a taxi or going by public transport differ in both cost and travel time. A method for comparing travel times between taxi and public transport is presented in this paper. A naive method for estimating travel times for taxi is implemented and evaluated for real trips in Stockholm. The method makes use of historic GPS data from taxis and constructs a database of observed speeds for separate roads. When estimating a new trip, the gathered observations are used to compute the new travel time. Public transport travel times are retrieved from an external API provided by the public transport services in Stockholm. The evaluation of the taxi travel time estimation shows that the mean error of the estimations is 14.8% and that most estimated times are underestimated. The average estimated travel time is 10% lower than the real time. The travel times can be computed in instant time which makes the solution suitable for usage on websites and applications. It is assumed that the method could be improved by taking more environmental factors into consideration and by using a more detailed digital road network.

Sammanfattning
Estimering av restid för taxi och kollektivtrafik
Contents

1 Introduction 1

2 Background 1
   2.1 Measuring traffic 2
   2.2 Map matching and path inference 3
   2.3 Travel time decomposition 4
   2.4 Travel time estimation 6
   2.5 Public transport estimation 7

3 Method 8
   3.1 Map matching and path inference 8
   3.2 Creating observations 12
   3.3 Travel time estimation 13
   3.4 Public transport estimation 14

4 Results 15

5 Discussion 18

6 Conclusion 20

7 Future work 20

8 References 21

Appendix A Additional results 23
1 Introduction

Knowing when it is beneficial to travel by taxi instead of waiting for the next bus is a challenge, since they differ in both cost and time. Most taxi companies do not offer an estimation of arrival time when you book your trip, and it might be hard to estimate it yourself, especially if you are not used to driving in that area. Public transportation, on the other hand, often offers an estimation since buses and trains follow a time table.

To enable such comparison, several approaches to estimate the travel time for a car will be reviewed and a method will be implemented and evaluated. Since public transport timetables can easily be accessed on a website and retrieved via an API (Application programming interface), this work will mainly focus on how to estimate the taxi travel times.

Many route-planning tools such as GPS-devices and websites often estimate the time based only on distance and do not consider congestion and other traffic conditions. Google maps [7], one of the bigger competitors in the area, provides real time estimations from tracked devices [1]. However, there is no known information on how the estimation is done and from what vehicles the information is gathered. Since taxis in Stockholm are allowed to use bus lanes estimations based on other types of vehicles may be inaccurate, especially during rush hours when bus lanes can have a significant impact on the travel time. To make an estimation for the right type of vehicle old GPS data gathered from taxis in Stockholm will be used to compute the estimated travel time.

2 Background

Research on travel time estimation is an old topic, which has experienced changes due to new technologies for gathering data. Many of the older devices for gathering data are stationary, which limits the possibilities to supervise big areas. This has resulted in much research on congestion, focusing on highways and freeways, which has been, and still is a big problem in traffic. GPS opened up for new areas of research, by enabling the recording of data on a whole road network, rather than a few chosen roads. This whole area of research touches many traffic-related problems, for example route planning and congestion control.

In the remainder of this paper, a data point from a GPS is known as a transmission, and continuous transmissions from the same vehicle form a trace representing a trip. A link or a segment refers to a road which may only have intersections in its beginning or its end. Figure 1 displays these concepts.

In order to make travel time estimations from gathered data several steps in the literature has been identified:

- **Gathering data** - The first step is to actually gather the data; to measure the traffic. This area has experienced great change in the last years and a short summary of how it can be done is given in the following section.

- **Map matching and path inference** - The measured data needs to be mapped to real roads in order to form the path that was travelled. The necessary
steps for this are called map matching and path inference. To match a trace of transmissions to real roads, a digital map is needed for knowing where the roads are positioned.

- **Travel time decomposition** - After matching the trip to the road network the time and distance of the trip needs to converted to observations of the travel time for the path. These travel times are distributed among the links in the path and stored as observations for each link.

- **Travel time estimation** - The observed travel times can be used for estimating the travel time of a new trip. The exact path of a new trip is usually unknown and the only known data is its start and goal coordinates, which means this step involves calculating the most probable path for the trip to know which links to use for the estimation.

These required steps are presented in the following sections, including a section about gathering information about public transport travel times. The main focus of this paper will be put on the last two steps: travel time distribution and estimation.

### 2.1 Measuring traffic

An old but commonly used technique for measuring traffic is loop detectors [14]. A loop detector is placed under a road and registers when a vehicle travels over it. Automatic License Place Recognition (ALPR) [4] scans the number plates of vehicles to see how fast they move between two cameras. Loop detectors and ALPR enables calculating the pace of the traffic flow to see when congestion emerges. The drawback of these techniques is that they are stationary, and therefore expensive to place on all roads, which makes them unsuitable for monitoring a whole road network. More commonly used nowadays is the GPS (Global Positioning System) technique, which is integrated into most smartphones. GPS makes
it possible to continuously gather data about the position of the vehicle while it is moving, and it is relatively cheap to use. The use of GPS is increasing as a consequence of the increased usage of smartphones. While stationary techniques suffer from immobility, GPS suffers from inaccuracy. High buildings and tunnels, for example, can make the GPS signal reflect itself on its surroundings, decreasing the precision of the reported position [6]. To avoid many of the problems with real traffic data, research done on traffic is often evaluated using simulated traffic data.

2.2 Map matching and path inference

Map matching is used for matching the GPS transmission to the most probable road segment for the reported position. To know where all existing links in the road network are positioned a digital map is needed. There are different levels of what data may be present in a digital map. The more advanced maps hold information about traffic lights, road signs, lanes and crossings for example. The simpler ones contain information about the speed limit of the roads. Digital maps over Stockholm from Lantmäteriet [15] can be accessed via Sveriges Lantbruksuniversitet [19], another alternative is to use OpenStreetMap (OSM) [16] which is an open source alternative. Common problems with digital maps are for example missing roads and faulty or missing speed limit data.

The step of binding map matched GPS transmissions together to form a path along the road network is known as path inference. Figure 2 shows the results of a trace after path inference. Many methods have been developed for these two steps, due to the importance of binding the transmissions together to form the most probable path. Research is often focused on mapping low frequency data since it is more challenging to map transmissions located further away from each other. Even closely placed transmissions may have several possible paths, see Figure 3. The number of possible paths between two transmissions grows with the distance between them. The method that will be used in this paper was developed by Koutsopoulos and Rahmani [17]. For each transmission in a trip, the N closest links are chosen to be candidate links for that transmission. Candidate links represent the possible links that the car emitting the transmission could have been positioned on when reporting its position. The permutations that can be formed by picking one candidate from each of the transmissions set represent possible paths taken by the vehicle. The most probable one of these permutations is used to represent the real trip. The authors suggest several ways to compute the most probable path, for example using the shortest path, or ratio between the length of the path to the length of the trace. To know the distance of a path the links used between the candidate links need to be known. The search algorithm A-star [8], known for finding the shortest path in graphs, is used for finding the links used between these candidates. The A-star algorithm uses a cost function to determine the cheapest path between a start and a goal point in graph. The cost function for A-star in Koutsopoulos and Rahmani [17] uses speed limit data to be able to make realistic assumptions; to allow valuing a highway higher than a small road in a residential area.

Koutsopoulos and Rahmani’s suggested method was evaluated and compared to three other methods, using real data from Stockholm. The results showed that
the suggested method outperformed the other methods, and identifies approximately 95% of the links correctly for when there is 30 seconds between transmissions.

2.3 Travel time decomposition

After having mapped the trace to the corresponding road segments, the measured travel time between the sequential transmissions in the trip has to be decomposed into travel time for the individual road segments used between the transmissions. There are several factors that complicate this task:

- Distance between transmission and segment - Due to the impreciseness of GPS the transmission will rarely be placed on the same coordinates as the roads. The transmission needs to be projected onto the segment to enable computing the travelled distance of the segment.

- Closely placed transmissions - When two transmissions are placed close to each other, the time and distance between them may be skewed due to imprecise positioning caused by the GPS.

- Speed limits - The path between two transmissions may contain roads with different speed limits. It is not guaranteed that the driver will increase the speed to reach the limit, and if that happens it is hard to tell on what segment that increase will take place.

- Pedestrian crossings and traffic signals - As with speed limits, it is not possible to know where between two transmissions the travel time is affected by red lights or pedestrians crossing the street.

Hellinga et al. [9] developed an algorithm for distribution of travel times based on the assumption that the travel time for a link consists of four parts: free flow travel time, stopping time, acceleration and deceleration, and delay due to congestion. They define free flow time as travelling in the same speed as the speed limit along the link. They introduce a congestion index to represent the ratio of congestion time to the sum of both the congestion time and the free travel time on the route. The algorithm is based on the assumption that the congestion of all links in a route is similar. They compute the probability of stopping, without information
about traffic control devices, by using the congestion index, as the probability of stopping tends to increase during higher congestion. Their algorithm proves to be most effective when the transmission frequency is around 60 seconds but it also works for lower frequencies. The algorithm is benchmarked against a simple distribution of time proportional to free flow time, which it outperforms.

Zheng and Van Zuylen [24] also tried estimating the travel times on separate links but used a three-layer artificial neural network (ANN). They compared their method to Hellingas’ using a simulated traffic network, and the ANN showed better results for a transmission frequency of 60 seconds. However, Zheng and Van Zuylen had access to GPS speed data, while Hellinga had not. They also tested their solution without using the speed data and showed that it only had a small impact, which they believed was due to the great variations in speed when travelling on urban roads.

Yuan et al. [22] takes another approach which eliminates the need to compute the travel times for all individual links. They use historical GPS data from taxis to give driving directions for the fastest path between two positions. They construct a landmark graph where the most frequently travelled links in the road network are represented as landmarks. An edge is added between two landmarks if a certain number of traces run between them. The travel time distribution for each edge is estimated using a clustering algorithm which takes the time of the day into consideration. The motivation for using landmarks is that creating landmarks does not require estimating the travel time for each link in the network, which is advantageous for sparse data and computationally efficient. To account for the different traffic conditions during weekends and weekdays, they use one graph for weekends and one for weekdays where the stored observations are created from

Figure 3: Multiple possible paths between two transmissions.
their respective type of day. They test the capabilities of using their algorithm for estimating travel times and the results show that the estimations are close to the real travel times.

### 2.4 Travel time estimation

To compute the estimated travel time for a whole trip, the travel times of the matched segments need to be concatenated. The following paragraphs will give a review of methods that cover both the creation of travel time observations, as well as the concatenation of them.

Zheng et al. [20] used a three dimensional tensor to model the travel time for separate road segments at separate times with satisfying results. They present a solution for concatenating the segments in an optimal way. In their work three problems concerning travel time estimation are discussed:

- **Sparse data** - Often there are road segments without historic data since these segments are not commonly used. To counter this problem, they assume that the travel time for road segments in the same context are alike. Bernard et al. [2] analysed correlations between link travel times and found that the correlation is high between links closely located to each other, and that the correlation decreases as the distance between the links increases. Zheng used distance as a factor when finding roads in the same context as well as the number of neighbours and additional environmental factors.

- **Concatenation of road segments** - To find the optimal concatenation they tried to minimise the risk of getting an inaccurate observation. They considered the number of observations and variance among the possible links to select the links with the lowest risk of being inaccurate.

- **Efficiency and scalability** - They concluded that if the solution should be usable for any road in a city which might contain many thousands of roads the solution needs to be able to access data for any road quickly.

Jenelius and Koutsopoulos [12] presented a statistical model whose parameters are estimated using maximum likelihood, which they used to estimate travel times with sparse, low frequency data. The model was used for calculating the travel time for links, assuming the travel time of a trip consists of running time along links and delay time at intersections. Spatial clustering of links was used to enable estimating the travel time for links without earlier observed travel time. The links were grouped into classes based on their characteristics, such as the type of way (for example primary or secondary road) until each class had a minimum number of links and observations of travel time.

Their results showed that there are positive correlations between the travel times of road segments. They found that certain conditions, such as weather, day of the week and time of the day had a significant impact on the mean and variation in travel time. However, rain had no significant impact on the travel times while snow made the travel time rise 1% per every 4 hours of heavy snowfall. They used data from taxis and concluded that occupied taxis generate around 6% higher travel time rates than free ones, which they believed depends on drivers cruising around slowly when looking for customers.
Chen and Chien [3] proposed a model for travel time prediction in real time using a Kalman filter [13], since this allows the state variable, which is travel time, to be continuously updated as more data is gathered. Two approaches were compared, a path-based and a link-based. For the path based, only observations from the beginning and end of a path are used for estimating travel times along the whole path. In the link based, the travelled time is recorded at each link. They argue that short interruptions in travel time from left turns for example may raise the average travel time for a link even though the current trip does not intend to do a left turn while the path based model would skip that observation for a vehicle that is not doing that left turn. Their results show that the path-based approach has a better performance than the link-based. The problem of having a small sample size was mentioned; being able to sample only 1% of every vehicle during a time interval had a slightly negative impact on their results.

Westgate et al. [21] estimated the travel times for ambulances using Bayesian data augmentation. They also presented two simpler solutions, the first one computes the harmonic mean of all observed travel times for each segment, and the other one assumes a log-normal distribution for each segment and uses maximum-likelihood estimation to estimate the parameters of the distribution. The advanced method outperforms the other two, which perform similarly, but they recommend the second simple solution if one wants an easy-to-implement solution, which is also suitable for larger data sets. Worth noticing is that their study was done with access to GPS speed data.

Hunter et al. [11] used an expectation-maximisation algorithm for estimating the distribution of travel times for links. They also add a latent variable to represent the path uncertainty between GPS transmissions. Their results show that the log-normal distribution yields better results than a Gaussian distribution. Emam and Al-Deek [5] proved the log-normal distribution to be good for freeways with loop-detector data but Hunter et al. shows it to be usable for GPS data and urban roads as well. The authors mention the fact that the correlation between link travel times should be taken into consideration.

When estimating the travel time of a new taxi ride, the only thing known is the start and the goal, opposed to when creating observations where the transmissions makes it possible to figure out what road segments were used. Zeng and Church [23] proved the A-star algorithm is useful for this.

2.5 Public transport estimation

The public transportation in Stockholm offers several API’s to access their services. The API Resrobot [18] can be used to send a request with coordinates for the start and goal and then returns a list of possible routes from nearby train stations or bus stops. The API allows setting several parameters that may affect the results:

- Walking speed - The speed used when calculating the time it takes to walk to the recommended station.
- Arrival or departure time - A time to specify when the traveller wants to leave or arrive. If no value is specified the API will return alternatives departing 10 minutes from when the request is sent.
• Means of transportation - Which means of transportation will be used or not used in the search, for example high-speed trains can be excluded while subway and buses are included.

3 Method

There are some limitations to what methods are suitable for estimating the travel time; the lack of GPS speed data and stoplights, for example, impacts the choice. One advantage that comes with the data set is that the GPS transmissions are collected with fairly high frequency, which makes path inference and travel time allocation easier. Unfortunately, as the data is sparse, it does not cover all roads at all times, which is a common problem that some papers has discussed [20, 12].

The sparsity of the data limits the possibility to let the estimation be affected by accidents and other real time events. With dense data, many vehicles can report prolonged travel times when accidents lowers the speed of a road. Using sparse data, a single observation of a prolonged travel time can easily be disregarded as an incorrect reading and will not have any visible impact on the travel time for that particular road.

The GPS data set consists of many million transmissions from taxi cars located in Stockholm. Each GPS transmission contains a latitude and longitude, a time stamp and a status telling whether the taxi is occupied or not. The majority of the transmissions are located in or around Stockholm but only data from Stockholm city will be used, to avoid making estimations of trips located in non-travelled areas. On average, the time between each transmission is 15 seconds. The transmissions span over six months in time, including winter months.

3.1 Map matching and path inference

The digital map from OSM [16] was used for creating a digital road network. It proved to have a higher precision than the map from Lantmäteriet, where many road’s coordinates were deviating from the true location of the roads and some roads were entirely missing. It also contained speed limit data for around 30% of the road segments. Links without speed limit data were assigned a default speed limit, set to 50 km/h.

When creating the digital road network the roads were split into smaller links so that all of the links only have neighbours connected to their ends. All links consist of a set of points, where each point has got coordinates for its position. One link has at least two points, marking its start and end. For a straight road, two points are enough, but for bent roads intermediate points are needed to describe the road. Links are neighbours if they share a point.

The GPS transmissions were put together to form trips. A trip consists of \( m \) consecutive transmissions from the same vehicles and is denoted as \( T \):

\[
T = [t_0 \ldots t_m]
\]

Transmissions from unoccupied taxis were ignored, since drivers tend to behave differently compared to when occupied, as noticed by Jenelius and Koutsopoulos [12]. Trips containing transmissions that are located too far away from
Figure 4: Two consecutive transmissions $t_1$ and $t_2$ with their candidate links marked. The candidates of $t_1$ are neighbours with the second candidates of $t_2$.

Each other in distance or time were removed since it will be more difficult to map those trips to the correct road segments. The maximum allowed distance between two points was set to 3000 meter and the time boundary was set to 70 seconds. An incorrectly mapped trip will result in a higher or lower travel time than in reality, which has a negative impact on the results. Transmissions located very close to the previous transmission were ignored; they are not needed to improve the precision and only complicate the path inference.

When matching a transmission to a link all links in a square centred on the transmission are selected as candidates, so that each transmission $t_m$ has a set $C_m$ of candidate links. The reason for using a square is motivated by the choice of data structure: in order to find these candidate links without having to inspect every point on the map a quadtree was used, which is a tree data structure. Each node in the tree represents a bounding box, which covers a geographical area. If the node is a leaf it is either empty or holds one or more geographical points positioned inside that bounding box. If the node is not a leaf it has exactly four children, one for each quadrant of the covered area. A capacity for how many points a leaf can hold is set, and if the capacity is reached, the boundary box will be split into smaller quadrants, see Figure 5. This enables finding coordinates closely located to each other efficiently. To find the candidate links, all of the points in a square with a given side length centred around the transmission are returned. All links containing those points are used as candidates. The sides of the square where points are sought are initially set to 200 meters.

Figure 4 shows the two transmissions $t_1$ and $t_2$ where $C_1$ consists of four links while $C_2$ consists of seven links. Each link in set $C_m$ is neighbours with all of the links in set $C_{m+1}$, the next transmission’s set. If the number of found links is lower than a certain threshold (in this case set to 4), the sides of the square are doubled to enable finding more candidates. If no candidates are found the trip will not be used for gathering traffic information. The risk of not finding candidates for a transmission is increased when the road map is missing roads, or when the position of the transmission is faulty. Faulty positions may occur when travelling through tunnels or close to high buildings and means that the reported position is far away from the true position of the vehicle.

When candidate links has been selected for all transmissions in a trace, the
most probable path between them was computed, which will consist of one candidate link for each transmission. The choice of path was based on a few criteria; the total distance of the path and the chosen candidate links’ distances to their transmissions. Rewarding candidate links that are closer to their transmissions was tested and showed to produce a more accurate matching of the path. Algorithm 1 shows the pseudocode of the path inference algorithm that was implemented. The algorithm is a modified version of A-star, using A-star itself for finding the distance of the road between two candidate links. Note that the candidate links are unique on link id and transmission number, since the same link can be a candidate to several transmissions.
**Algorithm 1 Path inference**

**Require:** \([C_0 \ldots C_m]\)
- \(\text{openSet} \leftarrow C_0\) {Adding all candidate links in \(C_0\)}
- \(\text{closedSet} \leftarrow \text{Empty set}\)
- \(\text{cameFrom} \leftarrow \text{Empty map}\) {For navigated candidate links}
- \(\text{goals} \leftarrow C_m\)
- \(gScore[C_0] = 0\) {Mapping link to cost for path}

**while openSet not empty do**
  - \(\text{current} \leftarrow \text{link in openSet with lowest fScore}\)
  - add \(\text{current}\) to \(\text{closedSet}\)
  - if \(\text{goals}\) contains \(\text{current}\) then
    - return reconstructPath(\(\text{cameFrom}, \text{goal}\)) {Backtracking path}
  - end if
  - for each neighbour of \(\text{current}\) do
    - if \(\text{closedSet}\) contains \(\text{neighbour}\) then
      - continue
    - end if
    - \(\text{distance} \leftarrow 0\)
    - if \(\text{neighbour}\) is not \(\text{current}\) then
      - \(\text{distance} \leftarrow \text{A-starDistance}(\text{current}, \text{neighbour})\)
    - end if
    - \(\text{newGScore} \leftarrow gScore[\text{current}] + \text{distance} + \text{neighbours distance to transmission}\)
    - if not \(\text{openSet}\) contains \(\text{neighbour} \parallel \text{newGScore} < gScore[\text{neighbour}]\) then
      - \(\text{cost} \leftarrow \text{manhattanDistance}(\text{neighbour}, \text{nextTransmission})\)
      - \(\text{cameFrom}[\text{neighbour}] = \text{current}\)
      - \(\text{neighbours fScore} \leftarrow \text{newGScore} + \text{cost}\)
      - \(gScore[\text{neighbour}] \leftarrow \text{newGScore}\)
      - add \(\text{neighbour}\) to \(\text{openSet}\)
    - end if
  - end for
  - end while

While Koutsoupoulos and Rahmani [17] suggested using speed limit data for the cost function in A-star, early tests of the implemented method showed that it worsened the results since only 30% of the links in the road network used contained that data.

The result of the path inference is a set of sub-paths between the selected candidate links:

\[ P = [p_0 \ldots p_n] \]

Each sub-path consists of a number of links \([l_0 \ldots l_x]\), where each link has the length \(l^d\). The total distance of the sub-path is denoted as \(p^d\):

\[ p^d = \sum_{i=0}^{x} l_i^d \]

Each path starts and ends at a transmission. The time stamps of the start and goal transmission for the sub-path \(p\) are denoted as \(p^s\) and \(p^g\). Some transmissions were positioned very close to each other and they shared one or several candidate links. That might lead to a path consisting of only one link, which also is used by the previous and the following path. The middle path was then removed to simplify the path and avoid splitting links unnecessarily. In Figure 6 path \(p_{n+1}\) would be removed, resulting in path \(p_{n+2}\) stretching between transmission \(t_{m+1}\) to \(t_{m+3}\).
Figure 6: Three sub-paths $p_n$ to $p_{n+2}$ between transmission $t_m$ to $t_{m+3}$, covering the links $l_x$ to $l_{x+2}$.

Figure 7: The arrows show the projections of the two transmissions. The rightmost transmission can not be projected straight onto the link which means the closest end point of the link is used for the projection.

3.2 Creating observations

After using the GPS traces to form trips on roads in the digital road network, the trips can be used for creating observations of real travel times since the distance of the trip is now known. To create an observation the travel time the time between the transmissions needs to be distributed among the travelled links. Initially the time between the transmissions in a trip was distributed proportionally to the travelled distance on the links between them, inspired by one of the local methods in Westgate et al. [21]. A speed observation, $obs$, was added for each link in $p_n$:

$$obs = \frac{p_n^d}{p_n^p - p_n^s}$$

Since the transmissions are not placed exactly on a link, a projection of the transmission is created for measuring what distance of the link was travelled. If the transmission is located where it cannot be projected in a straight angle onto the link, the most closely located end of that link is used instead, see Figure 7. The time is then scaled proportionally to the travelled part of the link.

The first simple implementation for distributing travel time generated many unreasonable speeds that were either too high or too low for a car. These speeds often occurred for shorter sub-paths, probably due to increased impact of errors in the GPS position for shorter distances. The solution to this was to use the
previous and following transmissions for computing the travel time in order to get a longer distance between the transmissions and lessen the impact of faulty GPS positions. The speed observation $obs$ is added to all links in $p_n$: \[ obs = \frac{\sum_{i=n-1}^{n+1} p_i^d}{p_{n+1}^d - p_{n-1}^d} \]

In the case of Figure 6 (ignore that the path $p_{n+1}$ would be removed), the sub-path $p_{n+1}$ would use the time between transmission $t_m$ and $t_{m+3}$ and the distance between $t_m$ and $t_{m+3}$ to compute the speed, which then is stored as an observation for the link $l_{x+1}$. The first and last sub-paths are left out intentionally, since their values can not be smoothed out due to missing surrounding paths and the map matching proved to be less accurate in the start and end of a trip. This occasionally resulted in unrealistic values which was countered by setting boundaries, as Westgate et al. [21] did. If the computed speed is higher or lower than the chosen boundaries then these boundaries are used instead of the computed value.

For each observation the hour and day were also saved, to be able to estimate correct travel times during rush hours or weekends, as in Jenelius and Koutsopoulos [12]. The days were divided into two groups, workdays and weekends, since the travel times within these groups turned out to be very similar, which Yuan et al. [22] also noticed. The hours are not grouped together due to significant differences in the observed speed. The direction of the trip is not taken into consideration, which means an observation of the travel time for a vehicle heading into the city centre will also be used for estimating a trip that is headed out of the city. Taking direction into consideration would result in needing more data for each link, as an estimation only can use observations from the current direction of the estimated trip. That would require more actions to counter lack of observations.

3.3 Travel time estimation

When computing the travel time for an upcoming trip an hour, a day and a start and goal position are given as input. The A-star algorithm was used to compute the shortest path between these positions, and the links in that path were used for estimating the taxi travel time. The speed observation for each link, $l_{obs}$, was used to calculate the total estimated travel time: \[ traveltime = \sum_{i=0}^{x} \frac{l_i^d}{l_{obs}^i} \]

To make the estimation affected by congestion and other time-dependant factors, only observations from the same hour and day type were used. To make up for links that has no observations, either from the current time or day type or none at all, several choices were considered: Zheng et al. [20] and Jenelius and Koutsopoulos [12] assumed links in the same context to have similar travel times and Bernard et al. [2] proved that links closely located to each other had correlating travel times. Due to the insufficient data regarding the surroundings of a link, only the number of neighbours, the speed limit and the distance of the link could...
be used for finding context-alike links. Using the link’s neighbours’ observations was tested but proved to worsen the results. The final solution was to let the first hand alternative be using the average of observations from the correct hour and day. If no such observation exists, the average observation from that link (regardless of day or time) was used. If no such observation was found, the max speed limit of the link was used to compute the travel time. All of the observations were scaled to fit the travelled length of the link, by using a projection as shown before, in Figure 7. In contrast to when creating the observations, scaling the travelled distance is only applicable for the first and the last link in a trip when estimating the travel time.

The first results showed that both long and short trips tend to be underestimated, but short trips were more underestimated than longer ones. To make up for that a penalty was added as a cost for accelerating and decelerating in the start and end of a trip, since it should have a greater impact on the shorter trips than the longer ones. This resulted in less underestimated trips and reduced the differences in correctness between short and long trip estimations.

The lack of data regarding the surrounding made it impossible to account for costs for traffic lights, crossing, and left turns. From the splitting of links when creating the digital road network, it is known whether two following links originally were one. That means there are no left or right turns included when moving between those links. This knowledge is taken advantage of when adding a cost for turns. If the current and previous link were not the same link from the beginning, a penalty of 10% of the observed travel time is added to each link. See Algorithm 2 for pseudocode of the travel time estimation.

```
Algorithm 2 Travel time estimation
Require: [links = l₀ . . . lₓ in trip]
travelt ime ← 0
previousLink ← null
for each link in links do
    linktime ← get observation speed from database
    if link to previousLink includes a turn then
        linktime += linktime * 0.1
    end if
    travelt ime += linktime
    previousLink ← link
end for
travelt ime += c {acceleration and deceleration cost}
return travelt ime
```

3.4 Public transport estimation

To retrieve the estimated time for the public transport the start and the goal position for the trip were sent in a request to the Resrobot API [18]. It returns the possible trips between the given points. At most five trips are returned, sorted by their departure time. The time difference between the departure and arrival time in the first trip was computed and returned. The answer includes the time it takes to walk from the start position to the suggested station, as well as the walking time between the last station and the goal position. Results containing high-speed trains or boats were excluded to avoid improbable alternatives. The
time from when the search is done until the time for departure was not included in the travel time, since the waiting time for the taxi was not included either.

4 Results

The travel time estimation for the taxi trips was evaluated with cross-validation. The observation database was built up using two thirds of the taxi data, and the tests were executed on the remaining one third of the data. The evaluated trips were first map matched in order to imitate the real path, and then the chosen links were used for the travel time estimation. The time difference between the first and last transmission was used as the real travel time of the trip.

The trips formed from the GPS data vary in distance travelled and cover all types of areas in Stockholm: residential areas, highways and inner city roads. There are trips from different hours and days as well as different seasons. The average trip is 10 minutes long and contains scarcely 40 transmissions. For distribution of trip times, see Figure A.2 in the appendix.

Figure 8 shows the estimated time and the real time for the taxi trips. The estimated travel times are generally lower than the real travel times and the estimates tends to vary more the longer the trip is. Figure 9 shows the frequency of the estimation ratios (estimated time divided by real time) for the matched trips. 100% means the estimated time and the real time are equal. While there are many estimates around 100% the mean and median estimation is approximately 90% of the real time, brought down by the high amount of lower and more spread out estimates. Table 1 shows mean, median and quartile values for the estimated time divided by the real time, and for the absolute error of the estimations.

<table>
<thead>
<tr>
<th>Estimated time</th>
<th>Real time</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>90.5%</td>
<td>90.8%</td>
</tr>
<tr>
<td></td>
<td>1st quartile</td>
<td>80%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3rd quartile</td>
<td>101%</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>Median</th>
<th>1st quartile</th>
<th>3rd quartile</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Real time - estimated time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Real time</td>
<td>Mean</td>
<td>14.8%</td>
<td>12.0%</td>
</tr>
</tbody>
</table>

There was no difference in mean or median estimation ratio during the weekdays or the weekends, however, there are differences during the different hours of a day. Figure 10 shows the mean estimation ratios (estimated time divided by real time). The estimations are more underestimated during rush hours. To see what impact the rush hours has on the link’s observations the average speed for all links during the day were plotted in Figure 11. The total average speed of the link’s observations differs between 33 km/h up till 50 km/h depending on the hour. There is also a great variation in speed between weekends and workdays. The average estimation ratio of the trips and the average speed follow similar patterns.
Figure 8: The graph shows the relation between estimated (y-axis) and real (x-axis) times. The dark line indicates where the value for a perfect estimation is.

Figure 9: This graph shows the frequency of the estimation ratios (estimated time divided by real time). The marked bar shows where the average and median is located.
Figure 10: Average estimation ratios during the different hours of the day.

Figure 11: The total average speed observation by hour for all links and the average during workdays and weekends.
There are around forty thousand links in the digital road map, and about 20% of them have less than ten observations. Approximately 40% of the links has no observations at all. 15% of the trips contained links that had no observations, in those cases the speed limit had to be used as an observation for the link without observations. Among these trips the average percentage of links with missing observations was 4%. No significant difference was found between the estimation ratios of these trips compared to the over all estimation ratios, even the quartiles were the same (results can be found in the appendix, see Figure A.1).

To know whether it is fair to use the first one of the trips recommended by the Resrobot API, their suggested alternatives were evaluated. The first trip was compared to the shortest one among the returned alternatives. The results showed that, on average, the fastest alternative was approximately 10% faster than the first alternative.

The speed of performing the comparison was also tested. On average it took about 1 second to compare the different alternatives, which excludes creating the database of observations, as it is not needed for every new search.

5 Discussion

The amount of different parameters in the map matching and travel time estimation allows many combinations which complicates finding the optimal settings for the solution. In the following paragraphs, the factors with the heaviest influence on the implementation and evaluation are discussed.

The evaluated trip’s paths were found with map matching, when in reality only the start and goal positions are known, and the path used for the estimation is found with A-star. This may have a negative impact on the results in the evaluation, since the map matching can give rise to errors. The A-star search, on the other hand, may not give the preferred way either, since the shortest is not always the most reasonable way. The lack of speed limit data limited the A-star algorithms’ heuristic and possibly the travel time allocation, since the time was only distributed proportionally to the links length, instead of letting the speed limit have an impact of the distribution.

As Chen and Chien [3] mentioned, there is a problem with having a small sample size as the behaviour of these samples may not represent the behaviour of the average driver on the road. The sample size might be small enough to allow different driver’s patterns have an impact on the results. In this case the purpose of the estimation is to estimate the travel times for a specific group of vehicles so at least the sample fits the purpose. The sample size led to another problems, which is non-existent observations for roads. It was shown that very few links had to use the speed limit as the estimation, which only happens when there are no observations to use at all, and many links only had a few each. The evaluation showed there was no significant difference in the estimation ratios of the estimations between the trips with observations for all links and the trips with links that were missing observations. This leads to the conclusion that a low percentage of missing observations have no big impact on the estimation. Since both the evaluated data set and the one used to build the observation database contains taxi data, there is reason to believe that the trips cover approximately
the same areas, giving the travelled links a fair amount of observations even if the
total number of links without observations seems high.

The estimation ratios appeared to differ during the different hours of the day
and the average speed for links follows the same pattern. The higher the average
speed was, the longer the estimated time became. Normally a higher speed would
indicate a shorter travel time but the results show the opposite effect. A reasonable
explanation is that the surrounding environment has a greater impact on the travel
time when there are many vehicles and persons on the road, for example left turns
will take more time and the driver has to stop for pedestrians more often. These
circumstances are not taken into consideration when estimating the travel time
and are most likely the reasons behind these correlating values.

No distinction is made among driving directions, which affects roads that ex-
perience congestion in only one direction. This results in the speed for both
directions being smoothed out, increasing the observed speed during congestion
and decreasing it during light traffic. This should affect highways significantly,
but the road network has got separate links for the lanes in highways and most
arterial roads. Distinguishing between directions would make using sparse data
more challenging, since only observations for trips in the right directions can be
used.

As can be seen in Figure 8 there is also a shortage of data on longer trips
and on very short trips. The estimations are more spread out for longer trips,
probably since they contain more road segments and therefore the risk of faulty
estimations is increased.

It is important to have in mind that there are several factors that may have an
instant effect on the real travel time, such as traffic lights and pedestrian cross-
ings. A travel time estimation can never be expected to be perfectly correct, and
giving an interval of a few minutes must be considered acceptable. As Jenelius
and Koutsopoulos [12] showed, snowfall for example had some impact on their
results and in this solution observations from different seasons were mixed, which
might have had an impact on the average travel times for some links. The penalty
added for turns proved to improve the results somewhat, however the majority
of the estimations are still underestimated which probably depends on missing
information about the surroundings (traffic lights, crossings, etc.). Since the solu-
tion does use real time data the consequences of accidents and road construction
work can not be accounted for.

Since a public API is used to estimate the public transport travel time, their
method for estimating the time is unknown. However, the buses and subways
have a schedule to follow and can therefore be expected to be on time. There are
also questions to how fair the comparison of these travel times is. The taxi will
probably take some time to get to the customer which cannot be accounted for at
the moment, since the location of the taxi is unknown. In the same manner the
public transportation travel time may be treated unfair if the person intending to
travel is not in a hurry. As mentioned, there was a difference in public transport
between the first and the quickest path among the suggested ones. In reality the
traveller may want to wait for the quickest arrival rather than leaving for a longer
trip.

19
6 Conclusion

To be able to compare taxi travel times with public transport times a naive solution was presented. A data set with coordinates from taxis was used to create a database of observations of travel times for the roads they travelled. These observations are then put together when estimating a new trip. The only required input for comparing a trip is its start and destination coordinates, the time and the day of the trip. The estimation for taxi travel time finds the shortest path between the start and destination and summarises the travel times of the links in the path, using the database of observations. It then adds a cost for acceleration and deceleration, and a cost for turns. The suggested method takes the hour of the day and type of day (workday or weekend) into consideration.

The solution was tested with cross-validation on a data set containing taxi trips from all over Stockholm. The evaluation of the travel time estimation showed that the travel times tend to be consequently underestimated with approximately 10% lower estimated time than real time. The estimated times were more underestimated during rush hours. It was concluded that it depends on that the surrounding environment factors that are unaccounted for. The mean absolute percentage error of the estimations was 14.8%.

To enable comparing taxi travel time with the travel time of public transport an API was used for retrieving travel time from timetables. Neither the waiting time for the taxi nor the public transport is included in the comparison. Both the taxi travel time and the public transport travel time are computed in almost instant time which makes the solution suitable for usage in websites or applications.

In this study the database of observations is only used for retrieving travel time for a single road segment but the data can easily be applied to more scenarios. It could be used in route planning tools, to compute the quickest road, instead of the shortest. Since the observations contain data about time and day the computed road could differ during rush hours or weekends, resulting in avoidance of the worst congestion. The data also makes it possible to analyse traffic behaviour, and see when and where congestion emerges. Too see more specific patterns for congestion on smaller roads driving directions needs to be considered.

7 Future work

Future work could focus on estimating the total time it would take for a specific taxi to drive to the customer and then to the final destination to be able to account for the unknown arrival time of the taxi. Then the time until the suggested departure with public transport could also be accounted for.

A better road map is recommended for improving the results. More detailed information about the surroundings could open up for using observations from other roads in the same context and the heuristic in A-star could be improved as well. Traffic lights and pedestrian crossings should be accounted for, which would make the estimation more realistic.
8 References


[16] OpenStreetMap https://www.openstreetmap.org


[18] Resrobot - Sök resa https://www.trafiklab.se/api/resrobot-sok-resa


A  Additional results

Figure A.1: This graph shows the frequency of the estimation ratios for trips that contains link that has no observations. The marked bar shows the average and median estimation ratio. The x-axis shows the estimated time divided by the real time for the trip.

Figure A.2: The graph shows how the real trip times are distributed.