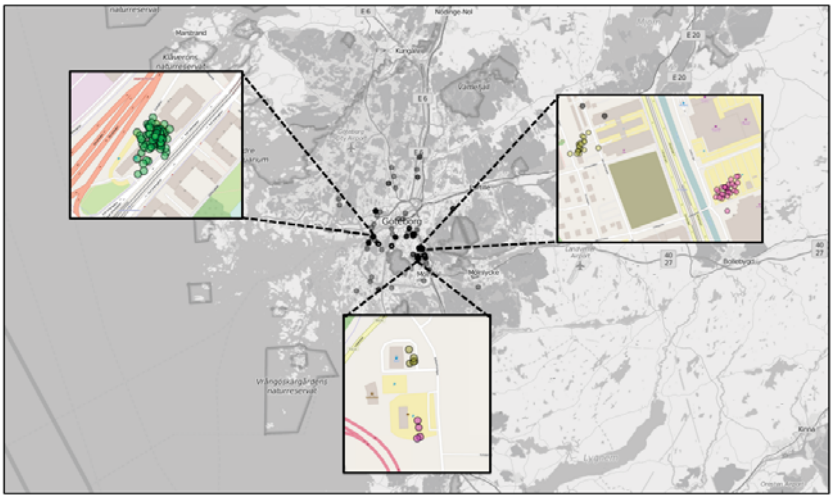

Clustering driver's destinations

- using internal evaluation to adaptively set parameters

CARL LEVIN, LTH

CHRISTOPHER HÅKANSSON, CTH

Dept. of Signals and Systems, Dept. of Automatic Control
CHALMERS UNIVERSITY OF TECHNOLOGY, LUND UNIVERSITY
Gothenburg, Sweden 2015

MSc Thesis
ISRN LUTFD2/TFRT--5995--SE
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2015

Cover: Map of Gothenburg area with clustered GPS data.

Abstract

With advanced navigation systems becoming ubiquitous in modern cars, the availability of detailed GPS data opens up new research areas in the fields of pattern analysis and data mining. By capturing the end-of-trip GPS points of each trip made by a driver, that driver's meaningful destinations could be identified. The knowledge of these destinations can be used for route prediction, which in turn can be used for optimizing the motor control to decrease emissions. It can also be used for developing functions for autonomous vehicles. In this thesis a way of extracting these meaningful destinations from GPS data using clustering algorithms has been developed and evaluated. The result is a clustering procedure consisting of 2 steps of clustering. First a pre-clustering to divide the data into subsets corresponding to smaller spatial areas. Then, a refining clustering step for which the parameter of the algorithm is adapted to each subset. Adaptively setting the parameter for each subset is done by testing a set of parameters and evaluating the results internally, with the Silhouette coefficient, and choosing the parameter giving the best evaluation score. The best performing configuration of our procedure, according to our external evaluation method, is in par with the performance of DBSCAN with a supervised choice of parameter setting. Further evaluation of data sets from different areas of the world are needed to draw strong conclusions of the developed procedures performance.

Keywords: GPS data, clustering analysis, DBSCAN, OPTICS, adaptive parameter, destinations, Silhouette coefficient, road distance.

Acknowledgements

This research was conducted at Volvo Cars Corporation which provided us with necessary equipment, software and support. We thank our supervisor, Niklas Åkerblom, and co-supervisor, Ghazaleh Panahandeh, from Volvo Cars who provided insight and assistance during the research. We thank our academic supervisor Giacomo Como from the Faculty of Engineering at Lund University for assistance with evaluation methodology, and for comments that improved the manuscript. We would also like to show our gratitude to Professor Martin Fabian at Chalmers University of Technology for taking the time to support us and provide us with good advice through the process of this research. Finally thanks to Professor Rolf Johansson, examiner at the Faculty of Engineering at Lund University, for his input on this master's thesis.

Contents

1
Introduction

1.1 Background

The emergence of modern techniques of data collection and capacity to store large quantities of data have given rise to an increase of research in the field of data mining. Data on its own is not of much interest, rather it is the information which can be extracted from the data that is useful, and the purpose of data mining is to extract useful information from data sets [17]. In the automotive industry, driving data such as GPS location, time, speed, motor torque etc. can be measured by a vehicle over time for purposes such as optimizing motor control to decrease emissions or developing functions for autonomous vehicles. The knowledge of *commonly visited destinations* for a vehicle can be extracted from such data, which is the main subject of this thesis. Having knowledge of commonly visited destinations and access to data which belongs to these destinations enables the ability to predict which destination will be visited next. This in turn gives the possibility to predict the route and ultimately optimize the motor control for the specific route. Moreover, additional data could be added and associated with specific destinations such as information regarding charging possibilities for electrical vehicles or availability of parking locations. Identifying destinations from the data in this case means grouping together data points which are assumed to belong to the same destination. Grouping data together based on similarity of attributes is a field of data mining referred to as clustering analysis.

1.2 Purpose

The purpose of the master's thesis is to develop a robust clustering algorithm to cluster driving data of unique vehicles into clusters corresponding to separate destinations for that vehicle. The data connected to the identified destinations can be used for example to predict the vehicles next destination and ultimately a trajectory to the predicted destination can be calculated.

1.3 Problem description

The main issues to overcome when facing a clustering problem regard finding or developing an algorithm which suits the data and the intended outcome. This consists of sub-problems. The intended outcome in this case is a clustering where

every cluster corresponds to a single unique destination. Hence when grouping points together into a cluster, it should be based on a similarity measure which best captures the essence of a destination. A sub-problem to solve is therefore to find a suitable similarity measure among the available data. The actual destination, i.e. where the driver goes after parking the vehicle, is not possible to know exactly only given vehicle data. Hence a destination in this case is rather a parking lot or an area where a driver can park when having an intended location in mind. Attributes of the data and data-distribution are explained in section 2.2 making the problem of finding a suitable algorithm more clear. The algorithm is thought to run in a vehicle which collects data, i.e. new data will be added to the data set every time the vehicle drives. Consequently, the algorithm should preferably handle sequential data. Moreover, the algorithm is intended to be embedded, without user input. Hence parameters settings for the algorithm must be either dynamically inferred or suitable for all the data. The goals set in the beginning of this thesis are embodied in the following list.

1.3.1 Goals

1. Evaluate and expand upon algorithms for clustering vehicle's destinations, with the goal of developing a robust method.
2. Analyze the available data to look for routine behaviour or other relationships that could be useful for clustering e.g. trajectory similarities for destinations.
3. Evaluate parameter settings for the algorithms and determine if they can be dynamically inferred.
4. Evaluate and test the performance of the developed algorithms on experimental data.

1.4 Delimitations

We do not aim to develop a route- or destination-prediction algorithm. We do not aim to label the different destinations with accurate real world names such as 'Home', 'Work' or 'School'. The results presented in this thesis are only from GPS data from vehicles in the south west of Sweden.

1.5 Method

As mentioned in the problem description an issue with most clustering algorithms is that these algorithms have parameters that need to be tuned to give the best result. The optimal parameter settings change with each data set and may also be different in different subsets. In an attempt to overcome this, a clustering procedure that adaptively sets the parameters for different subsets is developed. The procedure consists of a combination of two clustering algorithms. Hence, different combinations of algorithms will be evaluated. First a pre-clustering is done to distinguish smaller subsets of the initial data set. The purpose of this is to optimize a parameter setting for each subset. The initial segmentation is then refined using an algorithm

with an internal evaluation. The refining algorithm clusters the data points of each subset several times, using different parameter settings each time, and each result is evaluated by an internal evaluation measure. The refined clustering with the highest internal evaluation score for each of the subsets represent the final clustering. In addition to evaluating different combinations of algorithms, every combination is evaluated using two different similarity measures.

1.6 Disposition

Chapter 1 - Introduction
This chapter covers background, purpose, problem description, goals, delimitations and method.

Chapter 2 - Preliminaries
This chapter contains a presentation of the GPS data and theory regarding different clustering algorithms and evaluations used in this thesis.

Chapter 3 - Clustering Procedure
A presentation of all the steps in the developed clustering procedure. How the different algorithms were implemented in the procedure.

Chapter 4 - External evaluation
The external evaluation method that was used to generate comparable results is presented in this chapter.

Chapter 5 - Results
Visualization of clustering outputs. Results of external evaluation generated with different configurations of the clustering procedure. Caveats of algorithms and procedure.

Chapter 6 - Conclusion
The conclusions that were drawn with respect to the generated results.

2

Preliminaries

This chapter presents an introduction to clustering and clustering nomenclature, descriptions of the clustering techniques used in the thesis, a description of internal evaluation used in the clustering procedure and an explanation of external evaluation used to generate comparable results. For more detailed descriptions one can look into the references given in each section. This chapter also contains a presentation of the data.

2.1 Clustering analysis

As previously mentioned, the aim of clustering data is to partition it into groups in such a way that data points in the same group are more similar to each other, according to some *similarity measure*, than data points which are in separate groups. A crucial task when choosing clustering is therefore to identify a relevant similarity measure. Also recognizing characteristics of the data and data distribution is very important. For example if the data points appear to be sampled from separate Gaussian distributions, as the right image in 2.1, an algorithm which tries to fit the data to Gaussian distributions will perform well. If on the other hand the data is distributed in groups of seemingly random sizes and shapes and contains noise, such as the left image, then the algorithm which assumes Gaussian distribution will not perform well. This type of data is better partitioned with a density based algorithm as illustrated in Figure 2.2. A density based algorithm does not assume any shape of distribution. It only takes the density of points into account. *Unimodality* or rather local unimodality is often of interest, in this context meaning that the data has a single distinct peak (mode) of maximum density. Data with several modes is referred to as *multimodal*. In real world applications of data collection, the data is often *noisy* because the sensors used to capture the data pick up or create a signal which is not accurate. This requires the method of clustering to be able to distinguish noise from relevant data to avoid a skewed perception of reality. *Outliers* are data points which are considered to not be a part of a cluster even though they are not noise. In this report we will refer to outliers as noise as well.

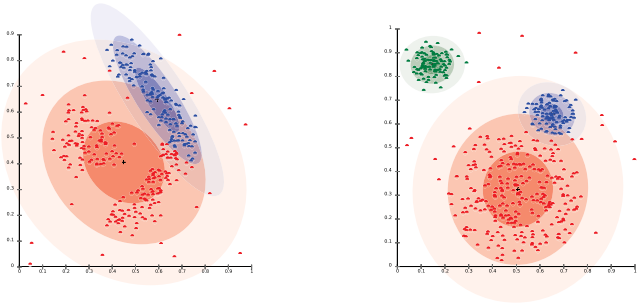

Figure 2.1: To the left is an image of a data set with non-convex properties and noise which has been clustered with an algorithm which assumes the data to have Gaussian distributions. To the right the data is more fit for such an algorithm. Image made by Chire, distributed under Creative Commons Attribution-Share Alike 3.0 Unported license.

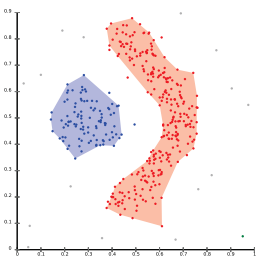

Figure 2.2: Cluster analysis with DBSCAN on a the same data set as the left image in Figure 2.1. Algorithm and data set are a perfect match for each other. The visualization was generated using ELKI.Image made by Chire, distributed under Creative Commons Attribution-Share Alike 3.0 Unported license.

More to take into account when faced with a clustering problem is that clustering algorithms have other traits which are relevant for the task. For example, a clustering algorithm can be *deterministic*, meaning that given a particular input will produce equivalent output. It can also be *incremental*, meaning that new data that is added to a data set which is already clustered can be handled without having to cluster all the data again. Computational complexity of the algorithm is often of importance as well.

2.2 Data

The data sets used consist of data points generated when the ignition of the vehicle is switched off. Data sets from different vehicles in the Gothenburg area with different numbers of points and different sets of destinations have been used. Each data point has the attributes: *Time*, *Latitude*, and *Longitude*. *Time* is when the stop

occurred and the *Latitude* and *Longitude* depict the position of the car in the global coordinate system.

2.2.1 Similarity measure

As mentioned in the Problem Description, a similarity measure which best captures the essence of a destination is desired. The spatial location is an attribute of the data where points belonging to the same destination will often be more similar than points which belong to different destinations. Hence the Euclidean distance between points is potentially suitable as a similarity measure.

Another proposed similarity measure is the driving distance between points. This is thought to even better capture the essence of a destination since it essentially includes the physical and law-bound limitations of the vehicle.

Since all the GPS coordinates have a time stamp the duration of the stop can be calculated. The duration, the time the stop was made or the leaving time of that stop could potentially be used as variables in a similarity measure. But incorporating a time variable together with a spatial distance in a similarity measure was thought to over complicate the measure increasing the dimensions of it. Also, no clear relationship could be found between time variables and unique destinations. The similarity measures which were chosen in this thesis were euclidean distance and driving distance.

2.2.2 Data distribution

Attributes of the data distribution are listed below.

1. The data is sparse.
2. The data forms denser areas which can take on a variety of convex or non-convex shapes.
3. The denser areas are generally not unimodal.
4. The data contains noise.
5. An arbitrary number of clusters can be formed over time.

Some of these characteristics of the data can be observed in Figure 2.3, Figure 2.4 and Figure 2.5. As can be seen in Figure 2.3 the data points are situated in a vast area. But in certain parts of that vast area the density of points is rather high. In those kind of areas, points belonging to distinct locations can be observed, as in Figure 2.4. Locations may also be situated in area where there are few other data points around, showed in Figure 2.5.

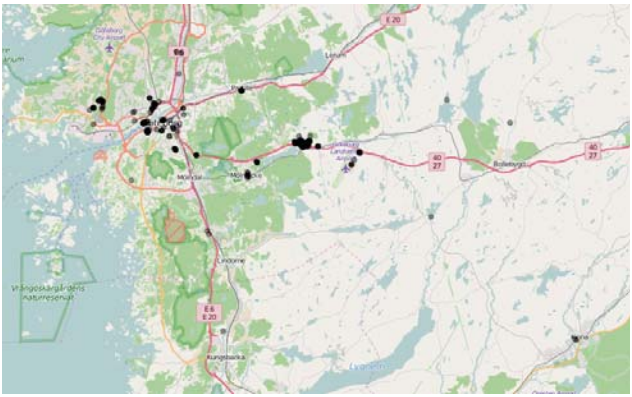

Figure 2.3: A data set consisting of parking events from one car. The data is generated from 6 months of driving.

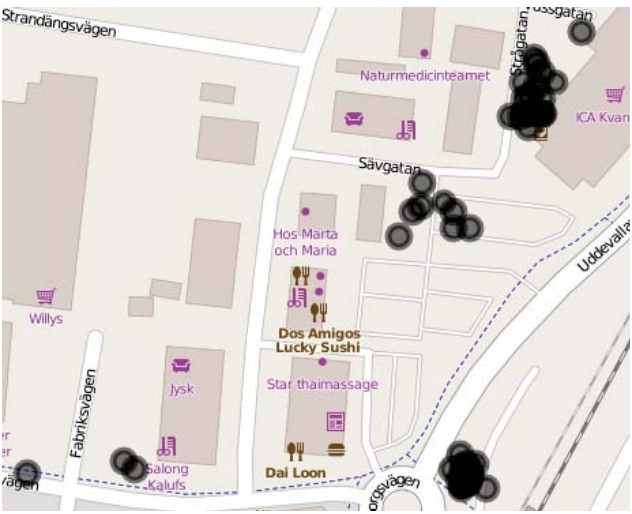

Figure 2.4: A subset of data containing several destinations.

Figure 2.5: A part of a data set where some points are close to each other but far from the other GPS points from the car.

2.3 Clustering algorithms

Three different algorithms were used and evaluated in separate configurations of the developed clustering procedure. Two of the algorithms are well known, DBSCAN (Density Based Spatial Clustering of Applications with Noise) and OPTICS (Ordering Points To Identify Cluster Structure), are presented in this section. The third algorithm was developed for this thesis and is based on minimum spanning tree segmentation and will be denoted *MSTC*. Hence a description of this concept is included here. Further specific implementation details follow in chapter 3. First a short motivation of choosing these algorithms is given.

2.3.1 Motivation of algorithm choices

Based on attributes of the data distribution and chosen similarity measures, density-based algorithms are a suitable class of clustering algorithms to evaluate since they can find clusters of arbitrary shape and size and do not require the number of clusters to be known. Still there are a variety of algorithms to choose from. In previous research on identifying destinations from GPS data corresponding to parking locations, a modified DBSCAN was used with euclidean distance as a similarity measure [15]. The results of this method shows high potential of using a form of DBSCAN for identifying destinations and for pre-processing the data by dividing it into smaller subsets.
The clustering algorithm OPTICS is also a density-based clustering algorithm. OPTICS is an expansion of the DBSCAN algorithm but with the difference that it has the ability to detect clusters in density-varying data whereas DBSCAN has a specific density threshold. Since the data is density-varying, the OPTICS algorithm was chosen to be investigated further. An other identified problem with DBSCAN

is that it can easily merge clusters which are close together if there are data points inbetween the clusters. A third algorithm which takes local peaks of density into account was developed to decrease the chance of merging destinations. This algorithm uses local minimum spanning trees of the data.

2.3.2 DBSCAN

DBSCAN is a clustering algorithm proposed by Ester, Kriegel, Sander and Xu in 1996 for doing what its name stands for: Density Based Spatial Clustering for Applications with Noise [5]. Given a set of data, DBSCAN will cluster together points that are close together in the feature space, i.e. where the density is high enough. Points in low density areas will be considered noise. This algorithm is widely used in theory and practice with its main advantages being the ability to detect clusters of arbitrary shape, that it can detect noise, that the number of clusters must not be known beforehand and that it has low complexity. DBSCAN takes two parameters: $minPts$, the minimum number of points to form a cluster and ϵ, a distance threshold. By setting the $minPts$ paramter to the value of 3 the DBSCAN algorithm becomes deterministic. The DBSCAN algorithms in this thesis are implemented that way. Since that is the case only the simplified version of DBSCAN with the $minPts$ parameter set to three will be presented in this section.

The algorithm examines the neighbourhood of each data point. The neighbourhood of a data point p meaning all the points within a distance lower than or equal to the value of the ϵ parameter from p. With the $minPts$ parameter set to three the data points can be classified into three classes. A data point is a core point if there are at least the amount of $minPts$ in its neighbourhood (including itself). A data point is a border point when there are not enough points in its neighbourhood to be considered a core point but there is a core point within the neighbourhood. A data point is noise when there are less than $minPts$ points within its neighbourhood. A point p is *density-reachable* from a point q if q lies within the neighbourhood of p and q is a core point. An unbroken chain of density-reachable points are considered a cluster. A simplified view of a cluster could be that the border points belonging to a cluster makes up the boundary of that cluster with the core points of that cluster situated inside of that boundary. An illustration of these definitions can be seen in Figure 2.6.

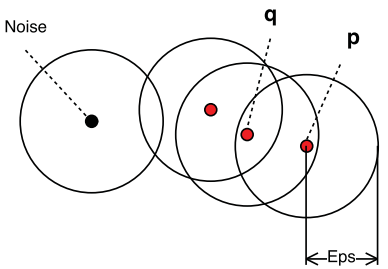

Figure 2.6: An illustration of DBSCAN with $minPts$ set to three. Point q is a core point. Point p is density reachable from q and has less than $minPts - 1$ neighbours, hence p is a border point. Along with the unlabeled point, q and p form a cluster.

2.3.3 OPTICS

OPTICS stands for Ordering Points To Identify the Clustering Structure and was first introduced in a paper with the same name in 1999 [3]. One major drawback DBSCAN has is that it has hard to find clusters in data with different data point densities. In the data encountered in this thesis it could be that DBSCAN would cluster two destinations together because of a point, belonging to no cluster, is spawned between the two destinations. OPTICS could be seen as an extended version of DBSCAN. The OPTICS algorithm introduce a $core - distance$, a $reachability - distance$ and a $clusteringorder$. In the implementation of the OPTICS algorithm the definitions of these concepts are: The $core - distance$ is defined as the smallest radius from a the point p so that p is considered as a core object. Objects are considered core objects under the same condition as core points in DBSCAN. The $reachability - distance$ is define as the maximum of the core distance of point p and the minimum distance between point p and the points which are ahead of point p in the $clusteringorder$. The $clusteringorder$ is updated when iterating through the data points and is prioritized with respect to the minimum pairwise distance from the the points which has been processed in the $clusteringorder$ to the points waiting to be processed. The algorithm is iteratively going through the data points and each data point is assigned a $reachability - distance$. In the original OPTICS algorithm the core distance is set to undefined if there are not enough neighbours within a specified maximum radius. This way the computation time could be lowered. This is not done in this clustering procedure since the data points will be divided into subsets before processed by the OPTICS algorithm.

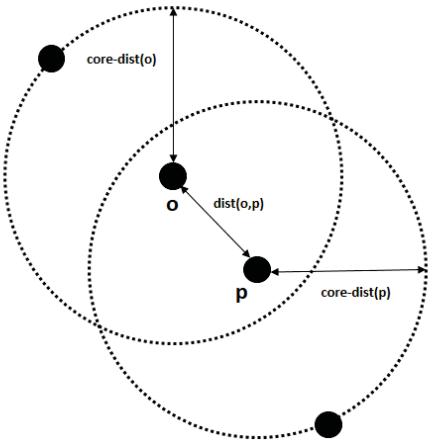

Figure 2.7: Illustrating the core-distances for $minPts = 3$

If the setup in Figure 2.7 is considered and point o is assumed to be the first point in clustering order. First the order is updated and since the point p is closest to o the $reachability - distance$ of p will be calculated next and since $core - distance(p) > distance(o, p)$ the $reachability - distance(p) = core - distance(p)$.
When the $reachability - distances$ of each point have been established the distances can be plotted as bars in a bar chart. An example of a reachability plot can be seen in Figure 2.8.

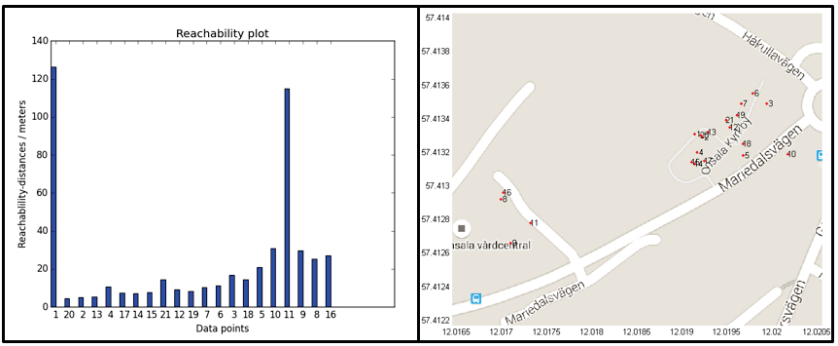

Figure 2.8: To the right is a reachability plot for a data set consisting of 20 points. The numbers on the x-axis is the clustering order. The left picture visualises the distribution of the data points represented in the reachability plot.

The valleys in the reachability plot correspond to the clusters of the data set. Data points that have small reachability-distances will have neighbours which are close and therefor should be clustered together. There exists different techniques to extract the clustering from the reachability plot. The methods used in this thesis, which deviates from the originally presented method, will be presented in section 3.3.

2.3.4 Minimum spanning tree segmentation

In graph theory, a spanning tree is a graph built up by a subset of a graph's edges such that all vertices (nodes) are connected and no loops exist. A minimum spanning tree (MST) is the spanning tree of a graph which has the lowest possible sum of weights [1]. Clustering using minimum spanning trees is convenient when the data can be separated by a relevant distance measure. Normally, the length of the edges indicate the weight relation between edges. Two well known algorithms to find the minimum spanning tree of an undirected graph are Prim's algorithm and Kruskal's algorithm. In Figure 2.9 an illustration of an MST is shown.

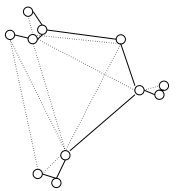

Figure 2.9: A minimum spanning tree (MST). Circles are vertices of the graph, lines are weighted edges of the MST and dotted lines are edges of the original graph which are excluded from the MST.

The MST-algorithm developed for this application is inspired by Felzenszwalb's algorithm for image segmentation which takes a graph as input and outputs several disjunct MST's [2]. In his work, the edges of the input graph are considered in order of increasing weight and the edges endpoint nodes are assigned to the same region (i.e. cluster) if no cycle is created and if the nodes fulfill constraints of similarity.

2.4 Evaluation of clustering

In this section a brief introduction to evaluation of clustering is given. The subject can be split into two types: *Internal* and *external* evaluation. In the clustering procedure developed during this thesis work an internal evaluation method is incorporated as part of the algorithm. External evaluation on the other hand is used as a tool to compare clustering results given by separate configurations of the clustering procedure.

2.4.1 Internal evaluation

Internal evaluation is done using only the clustered data itself and is typically based on computing some sort of ratio of within-cluster scattering (compactness) and between-cluster separation [7]. Since no external information is required, this evaluation can be done within the clustering procedure itself. Considering this, a clustering algorithm can be chosen to satisfy the internal evaluation as good as possible. However, the optimal clustering with regard to internal measures may not be the optimal clustering in the context of what the clustering is used for, which is why external evaluation is done.
In *An extensive comparative study of cluster validity indices* (2012), an extensive comparison of 30 different evaluation measures is done and a conclusion is that there is no measure which has a clear advantage over all others in all clustering cases [10]. The silhouette coefficient does however perform the best in many cases. Also, in *A comparison of clustering quality indices using outliers and noise* (2012), a comparison is done between different internal measures, but in this case specifically on data with noise and outliers. Here the measures are ranked in how they handle different data attributes and the silhouette coefficient is given the top ranking in handling noisy data [11]. It is relevant to mention that none of the evaluated measures handled noise particularly well. Based on the mentioned results, the silhouette coefficient was the first choice of internal evaluation method for our procedure. It is described in more detail in the following section.

2.4.1.1 Silhouette coefficient

The silhouette coefficient is an internal measure first introduced in 1986 by Peter J. Rousseeuw in [4]. The silhouette coefficient is a ratio type index which depends on the cohesion and separation of the clustering:

$$s(x_i) = \frac{b(x_i) - a(x_i)}{max\{a(x_i), b(x_i)\}} \tag{2.1}$$

In the common definition of the silhouette coefficient, x_i is an entity of the clustering, $b(x_i)$ the average distance to the entities in the nearest other cluster and $a(x_i)$ is the average distance to the entities belonging to the same cluster as x_i. Computing $a(x_i)$ and $b(x_i)$ as described will give the best results when the clusters have a circular form. When the clustering only consists of one cluster the $b(x_i)$ is somewhat unclear as there is no nearest cluster. When this type of clustering occurs $s(x_i)$ is set to zero, as it is also done in the original paper [4]. From the definition 2.1 it can be seen that $-1 \leq s(x_i) \leq 1$. To validate a whole clustering X the silhouette is defined as $s(X) = \frac{1}{n} \sum_{x_i \in X} s(x_i)$. The higher $s(X)$ the better the clustering is considered to be.

2.4.2 External evaluation

External evaluation evaluates the results based on external data. Often this would be the so called "ground truth" i.e. the actual classification of the data. In chapter

8.1 of *An Introduction to Information Retrieval* (2009) by C. D. Manning et al., cluster evaluation methods without ground truth are discussed. It is stated that one common way to deal with this is to let human "experts" within the area look at the data and define a *golden standard*, which is used as a ground truth. Recognizing that human judgements are idiosyncratic and variable, this is not considered to be a problem if the point of the clustering is to satisfy the needs of these same idiosyncratic humans [6].
Given a ground truth or a golden standard it is trivial to determine if the perfect clustering has been achieved or not simply by comparing the true classification with the clustering. The difficulty in the evaluation lies in determining how far from perfect an incorrect clustering is and how to value different traits of incorrect classification.
Most methods of evaluation consist of a type of normalized cost function which results in a value in the range [-1,1] or [0,1] where for example a higher value signifies better clustering. These values for different evaluation methods are called indices. Formal constraints, on external evaluation indices, formulated in [13] and extended in [14] suggest what properties are desirable for clustering in general. In these papers different indices are tested to see if they fulfill the constraints. This is in [14] done by evaluating two clusterings for each property, one clustering where the desired property is better fulfilled than the other. If the index is better for the clustering with the desirable property, it fulfills the constraint. The desirable properties are listed below:

1. **Homogeneity**, clusters should not mix objects belonging to different classes.
2. **Completeness**, objects belonging to the same class in the ground truth, should belong to the same cluster.
3. **Rag-Bag**. A Rag-bag cluster is a cluster containing objects from a mix of different classes. The constraint suggests that it should be penalized more to misplace an object into a pure (homogeneous) cluster than into a rag-bag cluster.
4. **Size vs. quantity**, a small error in a large cluster is preferable to a large number of small errors in small clusters
5. **Class size imbalance**. Misplacing an object from a small class into a cluster corresponding to a big class should be penalized more than misplacing an object from big class into a cluster corresponding to a small class.

A measure which consistently satisfies all of these constraints is not yet developed, to our knowledge.
Another alternative to using a golden standard or ground truth is to look at the performance of the clustering result in the context of its end goal. Our clustering procedure will be used in predicting the next destination when a driver starts the car, however, the prediction algorithm is still in the development stage and hence it's results can not be used for evaluation. In section 3.3.1 our method of evaluation is described.

3
Clustering Procedure

The clustering procedure consists of an initial DBSCAN which segments the the data set into smaller subsets. A distance matrix is calculated for each of these subsets. The similarity measure could either be the euclidean distance or the driving distance between the points. The subsets are clustered by a refining algorithm multiple times using different parameter settings. The refining algorithms used in this thesis are either DBSCAN, OPTICS or MSTC. The clusterings produced by the refining algorithm are evaluated internally using silhouette coefficient. The clustering with the highest silhouette coefficient is selected as the final clustering for that subset. The clustering of the different subset are put together and compose the final clustering of the data set. In Figure 3.1 a flow chart containing the different processes that the raw data will go through before resulting in the final clustering is illustrated.

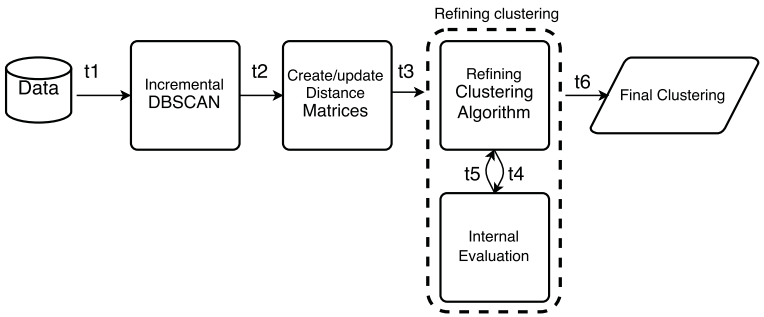

Figure 3.1: An overview of the developed clustering procedure. The variables $t1 - t6$ are transfers of data.

The following list describes what data is transferred at every transition.

t1 The raw data in the form $\{Time, Lat., Long., Y, X\}$.

t2 A pre-clustering of the data into smaller spatial regions. Each data point is assigned to a subset.

t3 A separate distance matrix for every subset.

t4 A refined clustering of a single cluster from $t2$ with a specific threshold.

t5 The internal evaluation index given by the current clustering of the current larger cluster.

t6 The refined clustering of every large cluster with a customized threshold.

Transitions $t4$ and $t5$ are done for every cluster given by $t2$, before $t6$ is enabled. The procedure is designed in such a way that it will work well with sequential data, as will be the case when implemented into a vehicle. Sequential, meaning one data point will be given at a time. The incremental DBSCAN handles this by only updating the subset where the new point belongs, hence not all data must be clustered again. Only the modified subset must continue through the rest of the flow illustrated.
In the following subsections an overview of the algorithms used and descriptions of the different steps of the clustering procedure are presented.

3.1 Incremental DBSCAN

The first step of the clustering procedure is the incremental DBSCAN algorithm. The algorithm divides the GPS points into smaller subsets to be able to customize thresholds of the refining algorithm for each area. The DBSCAN-parameter $minPts$ was set to 3 so that the algorithm is deterministic. The threshold parameter ϵ is by default set to 200 meters. This value has been confirmed to be big enough for capturing points which belong to the same destination in the same area, as well as being small enough to prevent very big areas. Big areas can potentially contain many destinations with varying cluster-attributes, obstructing the chance of assigning a suitable threshold. But in further research it might be interesting to examine the results generated with the ϵ parameter set to a higher value. This first DBSCAN step uses the euclidean distance in the 2D space as a distance-metric in every configuration of the procedure.

3.2 Distance matrix

The clustering output from the pre-clustering is fed into a function which computes a separate distance matrix for every subset. A visualization of the distance matrix can be seen in equation 3.1. The elements of the matrix, $d_{i,j}$, are the distances between point i and point j and n is the number of points.

$$\begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,n} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n,1} & d_{n,2} & \cdots & d_{n,n} \end{pmatrix} \tag{3.1}$$

In the clustering procedure it can specified which type of distance measure that should be used to generate the distance matrices. There are two options, the euclidean distance between the points and the driving distance between the points. To calculate the euclidean distance the longitude and latitude values of the points need to be converted to x- and y-coordinates. To calculate the driving distance an algorithm which can find the shortest path between two points in a road network is needed. Examples of these kinds of algorithms are Dijkstra's algorithm and A*. In

this thesis the values for the matrix were calculated using the service OSRM (Open Source Routing Machine) due to its high speed computation. However, at this moment in time, calculating the driving distance was not an available service. Instead the travel time (by car) is used as a similarity measure. The distance matrix could also be seen as an adjacency matrix representing a complete graph, the matrix is used by the MST-clustering algorithm to construct the minimum spanning tree.

3.3 Refining clustering

After the pre-clustering and the distance matrices for the different regions are calculated, the clusters and the matrices are fed into a refining algorithm one by one. The algorithms that were implemented and evaluated for this refining step are, OPTICS, DBSCAN and a clustering algorithm based on minimum spanning trees here referred to as MSTC.
In this refining step, each algorithm requires a parameter corresponding to a distance threshold. The clustering result is highly dependant on the choice of this threshold and using the same threshold for every pre-cluster subset will give poor results. By running the refining algorithm with a wide set of thresholds for every subset and evaluating each result using an internal evaluation measure, we can get a customized threshold for every region.

3.3.1 Internal Evaluation

The internal evaluation measure used in the clustering procedure is the silhouette coefficient, see section 2.4.1 for details. The data points in each subset are clustered by the refining algorithm using different parameter settings. The silhouette coefficient is then calculated for each clustering. The clustering with the highest score is then chosen as the best clustering of that subset. The values of the parameters to be evaluated were hard-coded into the algorithms. The parameter boundaries in this thesis were in the euclidean-metric-case set to 30 m minimum and 200 m maximum. Since the ϵ parameter is set to 200 m in the initial DBSCAN step there is no need to test a higher value. The lower boundary was set to 30 m since that was the lowest distance distinguished between two potential destinations. In the road distance case the upper boundary was set to 40 s since that was the time generated by OSRM when going from one side to the opposite side in the biggest parking-lot-destination. The lower boundary was as in the euclidean case set to the lowest road distance between two potential destinations, the value was set to 5 s. The refining algorithms all used the same parameter boundaries.
In the silhouette plot in Figure 3.2 the silhouette coefficient is plotted as a function of the parameter ϵ in the refining DBSCAN algorithm from the section 3.3. In this plot three plateaus can be recognized. The first between 30 and 40 meters, the second between 55 and 100 meters and the third starting at 110 meters. These plateaus correspond to different clusterings, the clustering corresponding to the first plateau contains three cluster, the clustering corresponding to the second plateau two clusters and the clustering corresponding to the thirds plateau contains one cluster. These clusterings can also be seen in Figure 3.2. Since the first plateau has

the highest value of the silhouette the algorithm chooses that clustering which was generated with the corresponding threshold.

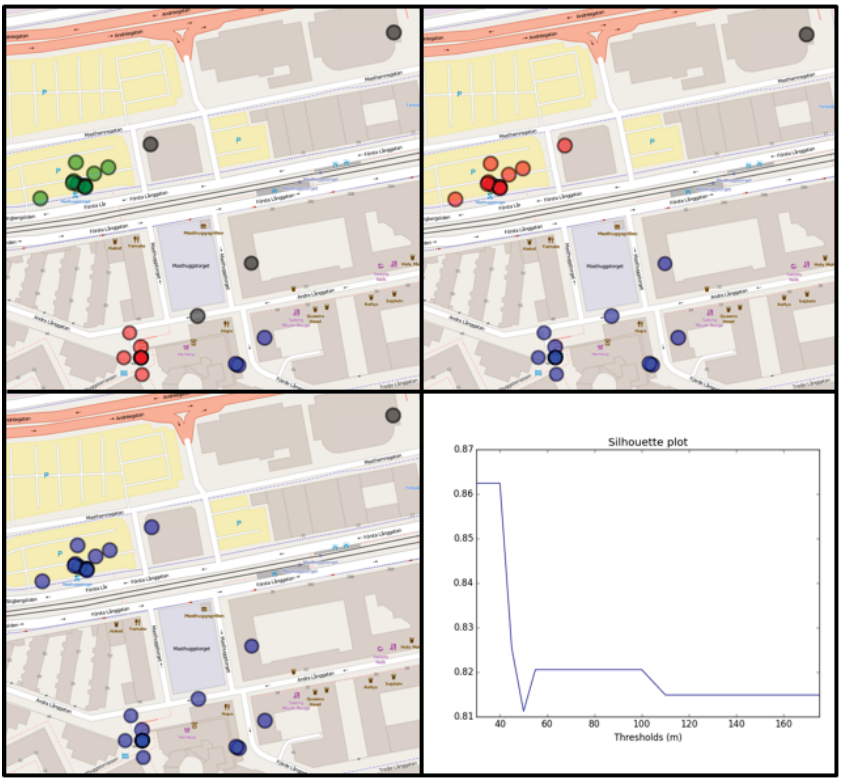

Figure 3.2: Clustering points using DBSCAN and different values of ϵ. In the top left overview the data set is clustered using $\epsilon = 30$, in the top right overview $\epsilon = 70$ and in the bottom left overview $\epsilon = 150$. The plot is showing the silhouette coefficient for the tested values of ϵ. The coloring of the points represent their cluster identity. Black points are noise.

3.3.2 DBSCAN

When DBSCAN is used as the refining clustering algorithm, the subsets are clustered multiple times using different values of the ϵ parameter. As mentioned earlier the parameter $minPts = 3$.

3.3.3 OPTICS - Cluster recognition

As stated in 2.3.3 there are several methods when it comes to extract clusters from a reachability-plot. The reachability-plot consists of reachability distances for the

points in the data set. The points are ordered on the x-axis according to the generated clustering order. The method used in this clustering procedure is basically to set a threshold and then examine the reachability-plot to find the points which have reachability distances that exceed the threshold and which points that have reachability distances below the threshold. The algorithm starts at the first point in the clustering ordering, i.e. the leftmost point in the reachability-plot. By default the reachability distance of the first point is set to a value above the threshold and is therefore considered as a potential start point of a new cluster. The remaining points are then iterated through with respect to clustering order until a point that has a reachability distance which exceeds the threshold is found. The data point in front of the threshold-exceeding point in the clustering order is considered as an end point and the threshold-exceeding point is consider as a start point of a new cluster. All points between, and including, the start point and the end point are counted. If the count exceeds or is equal to the parameter *minPts* a new cluster is generated, otherwise the points are considered noise points. A point is also considered noise if it is a potential start point of the cluster but the point next in the clustering order also has a reachability distance which exceeds the threshold. Some clustering for different thresholds can be seen in Figure 3.3.

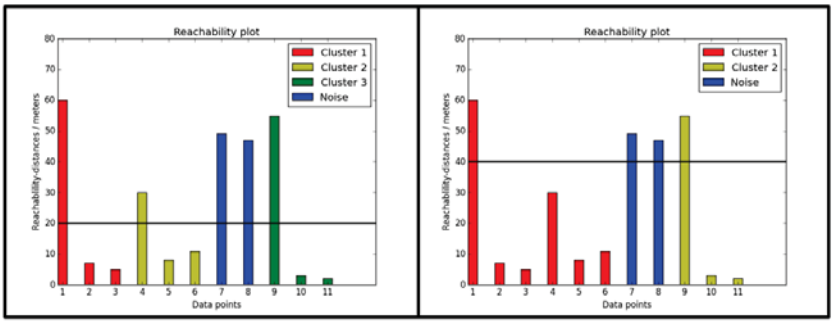

Figure 3.3: To the left a clustering of 11 data points using a threshold set to 20m. To the right a clustering of the same data points but using a threshold set to 40m. The *minPts* parameter was set to three in both of the clusterings.

The threshold is the parameter of the OPTICS algorithm which is updated between each clustering of a pre-clustered subset. The parameter *minPts* is set to 3 as in the initial DBSCAN algorithm.

3.3.4 Minimum spanning tree clustering (MSTC)

The MSTC uses both the complete graph (i.e. the distance matrix) and the MST of the graph. Similar to Felzenszwalb's segmentation algorithm, edges of the MST are considered in order of increasing weights. The end nodes of the edge are merged if the weight of the edge is below a threshold t. This threshold is the single variable input parameter to the algorithm. The merging has great impact on the result and

is explained shortly. This procedure continues until no edges with weights beneath the threshold are left. The nodes which have been merged together in the end will depict a new cluster. If any resulting clusters have less then 3 nodes they are considered noise. This algorithm can detect noise and does not need the number of clusters decided beforehand. The algorithm is summarized in the following steps.

1. Find smallest edge weight w_s in the MST
2. If $w_s <= t$, merge nodes connected to the edge.
3. Update MST (done in Merging function)
4. Repeat steps 1-3 until $w_s > t$.

3.3.4.1 Merging function

When two nodes are merged, it is meant that one of the nodes is removed from the MST, while its node ID is placed in the same bin as the others node ID. Also the node-count of the remaining node is incremented. Initially all nodes have a count of 1. The node which was not removed must be connected to the sub-tree which was previously connected to the removed node. The value of this new edge in the MST is taken directly from the complete graph. Hence the MST is updated and will be one node smaller for every iteration.

Choosing which node to merge to is done by counting which node has the most neighbouring nodes in the complete graph within a distance threshold. If the nodes have the same many neighbours, the node with the highest count is chosen. If they also have the same count, an arbitrary node is chosen. The radius of the circle around the point within which neighbours are found is $0.5 * t$. This merging function results in that the last standing nodes in the MST will correspond to local density peaks of the cluster. Something that neither DBSCAN nor OPTICS takes into account.

There are different ways to merge nodes and it has quite an effect on the results of the clustering. Also the function which decides which node to merge to has effect. A reason why the current merging function was chosen is to capture the local modality and in doing so, avoid merging destinations which have points in between them. This is illustrated with help of Figure 3.4, showing one iteration of the MSTC. In this iteration, n_1 and n_2 are both connected to the smallest edge d_s of the MST. If $t > d_s$, they will be merged. Since n_2 has more neighbours than n_1, n_1 is removed from the MST and n_2 is connected to the sub-tree previously connected to n_1. If $d_2 < t < d_1 + d_2$ then the resulting clustering would contain two clusters. DBSCAN, on the other hand, would have made a single cluster of all points using the same threshold ($\epsilon = t$).

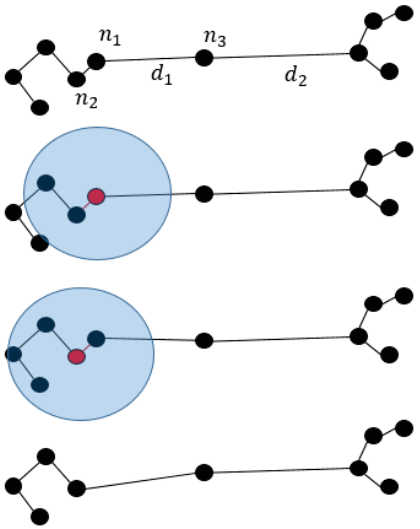

Figure 3.4: Illustrates an iteration of MSTC. $d1$ and d_2 are weights of the edges and $n1$ and $n2$ are the nodes connected to the edge with the smallest weight.

3.4 Implementation

The algorithms were implemented using Python. To make the structure of the clustering procedure easy to follow the whole procedure was implemented as a work flow in the program *Sympathy for Data*. Sympathy for Data is a framework used for data analysis. In Sympathy work flows are built which consists of nodes written in Python, this helps to visualize which steps that are used in the data analysis. The different steps of the clustering procedure were implemented as nodes in a Sympathy work flow. As can be seen in Figure 3.5 the sympathy work flow looks much like the overview of the clustering procedure in Figure 3.1. The Datasource node contains the data points of a vehicle. A Table node fits the data into a table which the incremental DBSCAN (IDBSCAN) node takes as input. The IDBSCAN node segments the data into subsets. The distance matrices of the subsets are calculated by a distance matrix node. The data points together with distance matrices are the inputs of the refining clustering node. The internal evaluation and the refining algorithm are implemented in the same node. The Map_clusters node visualises the clustering and the Evaluation node generates the external evaluation results. The external evaluation method will be presented in the next chapter.

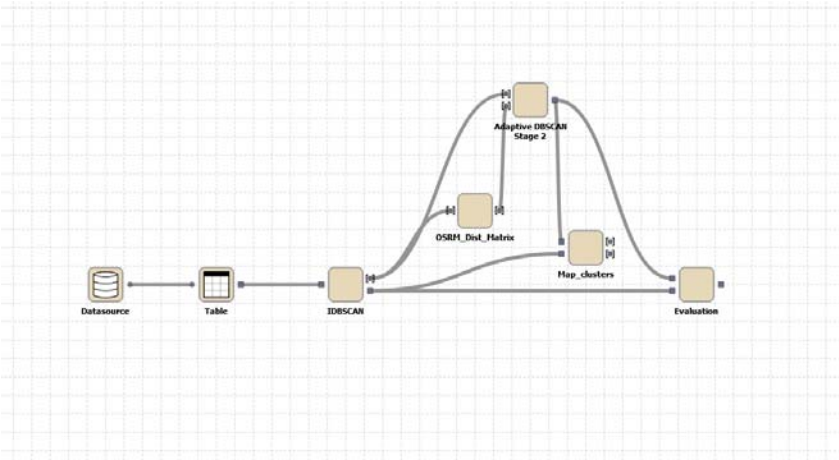

Figure 3.5: A picture of the work flow in Sympathy for Data used to cluster the data, visualize the clustering and evaluate the clustering.

4

Method of external evaluation

This chapter presents the external measure which was used to produce comparable results from different configurations of the clustering procedure. The chapter contains a description of how the golden standard was defined and a definition of the cost function developed to generate the results. First a short motivation of the developed method is given.

4.1 Motivation of the external evaluation method

Given that there is no ground truth for the actual classification available and that testing the results of the clustering in the context of its end goal was not possible at this point in time, we decided to define a golden standard. We are in this case the "experts" who define which areas correspond to a destination. There are many existing external measures for when a gold standard exists which perform differently regarding the desirable properties listed in section 2.4.2. Many of which are evaluated in [13] and [14]. Instead of trying to find one which satisfied what we want out of the evaluation we decided to formulate our own cost function which is specific to our goal. The goal of our external evaluation is to compare the resulting clusterings of applying different algorithms in the refining step of our procedure. Also to compare the results for each algorithm using two different similarity measures. The resulting cost values from our evaluation method is meant to be a strong indication of which performs best but can not be blindly accepted without discussion. The main advantage of our evaluation is that it will simplify this discussion by outputting separate costs for performance of *homogeneity*, *completeness* and *wrongly classified noise* as well as show the contribution to the total cost for each area distinguished by the pre-clustering. This speeds up the process of further manually analyzing the map where the clustering is not as desired.

4.2 Golden standard

To define a golden standard, the procedure is that all existing gps points for a car are plotted on a map. For this batchgeo.com was used, where longitude and latitude of points can be imported from an Excel file and mapped with ease. The map was viewed in satellite view for detailed understanding of the surroundings. A parking lot is distinguished as a destination, hence all points on the same parking lot will be clustered together. A shopping mall for example may have different parking lots on each side of the mall. Even though the driver's intention is to go to the mall

regardless of which parking lot is chosen, the parking lots will be distinguished as different destinations. Since the precision of GPS data can vary, points that are for example on a grass field just outside a parking area are assumed to belong to that parking area.

Many points however are not on or close to parking lots. In these cases definition of the golden standard is based on logical thinking. For example destination points belonging to a house with one driveway. The spreading of such an example can be seen in Figure 4.1. In these kinds of cases the parking area can be defined as a circle with the driveway as its centroid. Regarding the point which is closer to the neighbours driveway in 4.1, it is highly unlikely that a driver would take the car to their neighbour and hence it is classified the same as the rest.

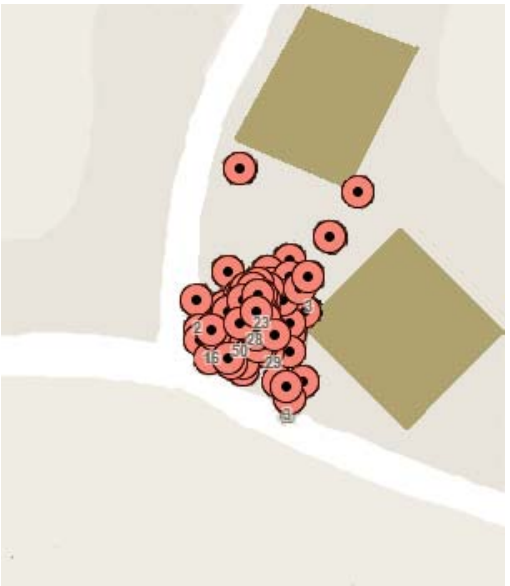

Figure 4.1: Data points belonging to a home residence.

As can be seen in Figure 4.2 it is not always clear how to classified the GPS points. The question is which parking areas should the points on the green area be assigned to? In this case since there is a parking lot to the left and just a road to the right the most logic thing would be to assign the data points to the parking lot. When there is this amount of data points situated on a non-drivable area one may suspect that the appearance of the road network has changed since the satellite image was taken.

Figure 4.2: Scattered data points.

4.3 Cost function

The whole road map $\{A\}$ is divided into small subareas $A = \{A_1, ..., A_m\}$ where A_i is a destination defined in the golden standard and m is the total number of identified destinations. Only areas with more than 3 points will be defined. Areas with noise can be seen as empty sets. If $x \in A_i$ where x is a data point, then $A(x) = A_i$. An example of the road map division can be seen in Figure 4.3. The cost function is based on pairwise comparisons between the partitions A and C, where $C = \{C_1, ..., C_k\}$ is the partitioning given by the clustering algorithm and k is the number of clusters found. The cost function is formulated in equations 4.1 and 4.2.

$$p(x_i, x_j) = \begin{cases} 0, & \text{if } A(x_i) = A(x_j) \text{ and } C(x_i) = C(x_j) \\ \alpha_1, & \text{if } A(x_i) \neq A(x_j) \text{ and } C(x_i) = C(x_j) \\ \alpha_2, & \text{if } A(x_i) = A(x_j) \text{ and } C(x_i) \neq C(x_j) \\ 0, & \text{if } A(x_i) \neq A(x_j) \text{ and } C(x_i) \neq C(x_j) \end{cases} \tag{4.1}$$

$$\Psi(x) = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} p(x_i, x_j), \quad x \in \hat{X} \tag{4.2}$$

Where α_1 and α_2 are positive constants, n is the number of points in the data set (excluding noise points from the pre-clustering step), $C(x)$ is the data points' cluster ID. and $\hat{X}$ is the set of data points excluding the points which are assigned as noise by the refining algorithm. The cost function Ψ increases with the number of data points that are clustered wrong. The clustering can be penalized in two different

ways. Either points that are located in different parking locations are clustered into the same cluster or that points which are belonging to the same parking location are clustered into different clusters. Where assigning points that belong to a parking location as noise points is a special case of the latter. An example of this can be seen in Figure 4.3.
Noise points generated from the clustering do not affect Ψ. But if a noise point is found in a parking area, we want our evaluation to capture this. Therefore Ψ_{noise} is introduced in **??**. Since noise points are not assigned to a cluster, their cluster ID. is set to -1.

$$p_{\text{noise}}(x_i) = \alpha_3, \quad \text{if } A(x_i) \neq \emptyset \text{ and } C(x_i) = -1 \tag{4.3}$$

$$\Psi_{\text{noise}}(x) = \frac{1}{n}\sum_{i=1}^{n} p_{\text{noise}}(x_i), \quad x \in X \tag{4.4}$$

The constant α_3 is a positive constant. Here X is the set of data.

Figure 4.3: An illustration of a road map division. The two areas are recognized parking locations. The nine points are assigned to two different clusters, one with six data points and one with three.

If the cost function of the clustering in Figure 4.3 was to be calculated it would be equal to: $\Psi_{\text{Total}} = \frac{2}{9*8}(3\alpha_2 + 3\alpha_2 + 3\alpha_2 + 3\alpha_1 + 3\alpha_1 + 3\alpha_1) = \frac{\alpha_2}{4} + \frac{\alpha_1}{4}$.
The contribution from α_1, α_2 and α_3 terms will be different for different clustering algorithms. In the result section the α_1, α_2 and α_3 costs will be presented separately as well as the total cost.

5

Results

This chapter presents visualizations, the external evaluation, computing time and caveats of the different clustering algorithms. The results were generated using the setup presented in Chapter 3 and the parameter boundaries described in section 3.3.1.

5.1 Clustering visualizations

Some visualizations of destinations discovered by the clustering procedure are presented in Figure 5.1 - 5.4. The points in each of the figures belong to one subset generated by the pre-clustering. As can be seen the refining clustering chooses different thresholds for different subsets and can handle subsets of varying character. Figure 5.1 shows a case when many separate destinations of varying shapes are identified within the same subset as well as noise. Figure 5.2 shows 2 identified destinations and no noise. Figure 5.3 shows a case when the driving time is used and successfully separates two destinations that are on different sides of a railroad track. Figure 5.4 shows a subset when a single cluster is identified along with 2 noise points.

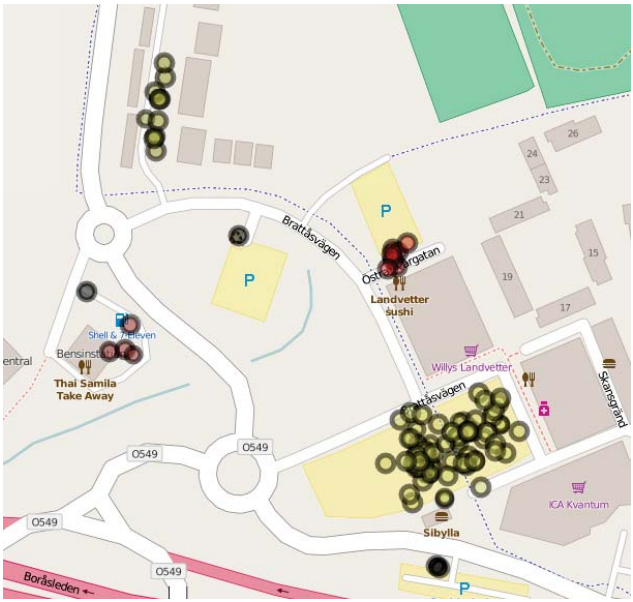

Figure 5.1: A clustering generated using DBSCAN as refining algorithm and the euclidean distance as similarity measure. Here the parameter ϵ was chosen to 40 meters by the internal evaluation method.

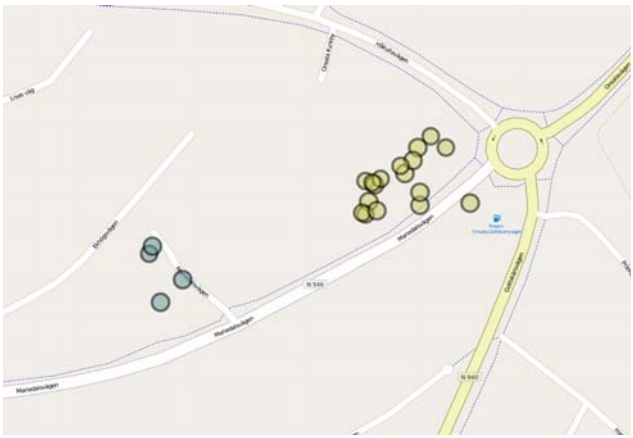

Figure 5.2: A clustering generated using DBSCAN as refining algorithm and the euclidean distance as similarity measure. Here the parameter ϵ was chosen to 35 meters by the internal evaluation method.

Figure 5.3: A clustering generated using DBSCAN as refining algorithm and the driving time as similarity measure. Here the parameter ϵ was chosen to 7 seconds by the internal evaluation method.

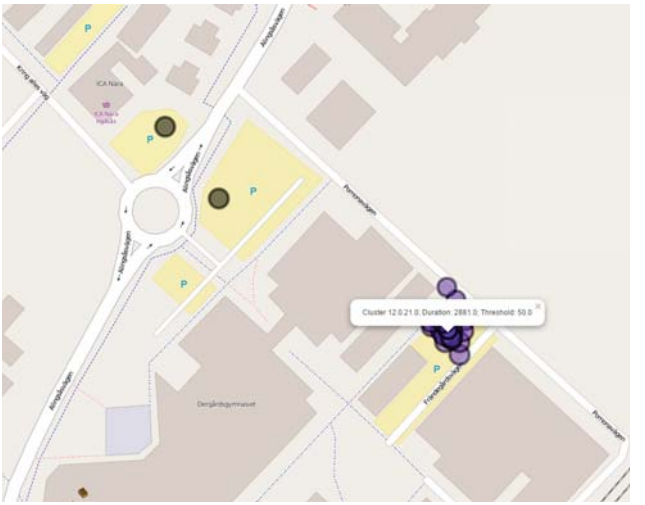

Figure 5.4: A clustering generated using DBSCAN as refining algorithm and the driving time as similarity measure. Here the parameter ϵ was chosen to 5 seconds by the internal evaluation method.

5.2 Cost function results

The cost function parameters α_1, α_2 and α_3 were set to 100, 100 and 1 respectively. The *Merge* penalty presented in tables 5.1 to 5.6 is the contribution from the α_1 terms from equation 4.1 while the *Separation* penalty is the contribution from $alpha_2$ terms. The noise penalty is denoted as *Noise intrusion*. Note that Noise intrusion is not directly comparable with the other values as it is calculated differently. It is a percentage of the points in the set which have been classified as noise when they should not have been, as can be seen in equations 4.3 and 4.4. Note also that the values of *Separation*, *Merge* and *Total cost* are not directly comparable between different data sets, as the value highly depends on the number of points in the set and the definition of the golden standard. The configuration column shows which similarity measure and which algorithm was used.

Configuration	Separation	Merge	**Total Cost**	Noise intrusion
Euclidean-DBSCAN	0.0320	0.4002	0.4322	0.0306
Euclidean-OPTICS	0.0792	0.4002	0.4794	0.0333
Euclidean-MSTC	0.8927	0	0.8927	0.0386
D.time-DBSCAN	0.0309	0.1548	0.1857	0.0599
D.time-OPTICS	0.1108	0.1548	0.2656	0.0599
D.time-MSTC	1.5911	0	1.5911	0.0639

Table 5.1: External evaluation results for data set of vehicle 1.

Configuration	Separation	Merge	**Total Cost**	Noise intrusion
Euclidean-DBSCAN	0.1357	0.0136	0.1493	0.0246
Euclidean-OPTICS	0.1671	0.0136	0.1807	0.0421
Euclidean-MSTC	0.1486	0.0154	0.1640	0.0351
D.time-DBSCAN	1.2395	0.0074	1.2469	0.0596
D.time-OPTICS	1.3702	0.0074	1.3776	0.0632
D.time-MSTC	1.3702	0.0074	1.3776	0.0544

Table 5.2: External evaluation results for data set of vehicle 2.

Configuration	Separation	Merge	**Total Cost**	Noise intrusion
Euclidean-DBSCAN	0	0.4380	0.04380	0.0306
Euclidean-OPTICS	0	0	0	0.0174
Euclidean-MSTC	0	0	0	0.0174
D.time-DBSCAN	0.3084	0.0219	0.3303	0.0377
D.time-OPTICS	0.3741	0.0219	0.3960	0.0464
D.time-MSTC	2.7654	0.0152	2.7806	0.0377

Table 5.3: External evaluation results for data set of vehicle 3.

Configuration	Separation	Merge	**Total Cost**	Noise intrusion
Euclidean-DBSCAN	0.0127	0.0127	0.0254	0.0357
Euclidean-OPTICS	0.0317	0.0127	0.0444	0.0552
Euclidean-MSTC	0.0438	0.3389	0.3827	0.0390
D.time-DBSCAN	0.0523	0.4097	0.4620	0.0422
D.time-OPTICS	0.1357	0.3373	0.4730	0.0536
D.time-MSTC	0.1088	0.3373	0.4461	0.0617

Table 5.4: External evaluation results for data set of vehicle 4.

Configuration	Separation	Merge	**Total Cost**	Noise intrusion
Euclidean-DBSCAN	0.0125	0.1685	0.1810	0.0259
Euclidean-OPTICS	0.0446	0.1685	0.0213	0.0330
Euclidean-MSTC	0.0462	0.1022	0.1484	0.0295
D.time-DBSCAN	0.0270	0.1328	0.1355	0.0307
D.time-OPTICS	0.0593	0.1328	0.1921	0.0307
D.time-MSTC	0.1493	0.1721	0.3214	0.0318

Table 5.5: External evaluation results for data set of vehicle 5.

Configuration	Separation	Merge	**Total Cost**	Noise intrusion
Euclidean-DBSCAN	0	0.2104	0.2104	0.0562
Euclidean-OPTICS	0.0559	0.2104	0.2663	0.0562
Euclidean-MSTC	0.0138	0.1835	0.1973	0.0580
D.time-DBSCAN	0.2913	0.0710	0.3623	0.0725
D.time-OPTICS	0.4044	0.0710	0.4754	0.0725
D.time-MSTC	0.3248	0.0710	0.3319	0.0725

Table 5.6: External evaluation results for data set of vehicle 6.

For the euclidean similarity measure, on average the separation-part of the total cost for MSTC, OPTICS and DBSCAN are 55 %, 44 % and 31 % respectively (Not including car number 3 since the total cost for the MSTC and OPTICS were 0). Which means the average cohesion parts of the cost function for MSTC, OPTICS and DBSCAN are 45 %, 56 % and 69 % respectively. Compared to the other algorithms the MSTC algorithm has its weakness in separating points into more clusters then what is optimal. The DBSCAN algorithm has its weakness in clustering points which don't belong to the same destination. The OPTICS algorithm is somewhere in the middle compared to the other algorithms. The three algorithms perform about the same when it comes to incorrectly assigning points as noise at about 3-4% of the points.
For the driving time as similarity measure, on average the separation-part of the total cost for MSTC, OPTICS and DBSCAN are 78 %, 63 % and 54 % respectively. The average cohesion parts of the total cost for MSTC, OPTICS and DBSCAN are

then 22 %, 37 % and 46 % respectively. The same weaknesses of the algorithms in comparison with each other can be seen for using driving times as for using euclidean distance. But the separation cost fraction increases for all the algorithms. The most probable reason for that is that the generated distances between certain points are erroneous, see section 5.5. The Noise intrusion is also generally higher for the driving time results, this most probably also depends on erroneous driving times generated with OSRM. The noise intrusion here is around 4-7%

5.2.1 Summary of cost function results

Table 5.7 gives a simplified summary of the cost function results by showing how many times each configuration performed the best w.r.t. lowest total cost. The best performing refining algorithm is DBSCAN and the best performing similarity measure is the euclidean distance.

Algorithm	Euclidean	Drive Time
MSTC	2	0
DBSCAN	2	2
OPTICS	1	0

Table 5.7: Cost function summary. Shows how many times each algorithm/similarity measure combination gave the lowest Total cost. OPTICS and MSTC performed equally well for one data set, explaining why the table values sum to seven.

5.3 Cluster findings

The cost function results are good for comparing some of the different qualities of the algorithms, but there are other qualities too take into account. Another measure to compare is how many of the destinations, identified by the golden standard, are found. Table 5.8 presents the fraction of the golden standard-destinations found by the different algorithms for the data from vehicle number 5. The data set consists of 911 data points.

	MSTC Euclidean	DBSCAN Euclidean	OPTICS Euclidean	MSTC OSRM	DBSCAN OSRM	OPTICS OSRM
Frac:	28/31	27/31	25/31	28/31	28/31	28/31
Total:	28	27	25	33	32	30

Table 5.8: The table shows how many meaningful destinations, identified in the golden standard, that the refining algorithms found. The Frac-values are the fraction of meaningful destinations found. The Total-values are the total number of destinations(clusters) found by the refining algorithm.

All the configurations show similar performance, identifying most of the destinations. The MSTC algorithm finds most destination but splits up the data the most in the

euclidean case. When using the driving time as similarity measure the algorithms find the same amount of destinations and once again the MSTC algorithm segments the data into the most number of clusters. The number of clusters found increases for all the refining algorithms going from euclidean to driving time as similarity measure.

5.4 Run times of the algorithms

The implementations of the algorithms have not been optimized, since that was not a goal of the thesis. Therefore we can not draw any conclusions from the run time results. For example the theoretical complexity of DBSCAN is lower than for OPTICS. But the run time of our DBSCAN implementation is longer compared to the run time for the OPTICS implementation. The run times of the current implementations of the algorithms for three different datasets can be found in Table 5.9.

Algorithm	Run time 1	Run time 2	Run time 3
MSTC	46.197 s	5.138 s	3.254 s
DBSCAN	8.018 s	2.670 s	1.482 s
OPTICS	4.772 s	1.580 s	0.699 s

Table 5.9: Run time of the refining algorithms when run together with the internal measure for three different data sets. The data sets have 792, 626 and 387 points from left to right respectively.

5.5 Caveats of Algorithms and procedure

From inspection of the mappings of clustering results together with the external evaluation, some typical faulty behaviour for the refining algorithms have been identified. An explanation of this behaviour and why it occurs is given below. Also caveats of using driving time as a similarity measure are explained.

5.5.1 OPTICS

Sometimes the output of the OPTICS algorithm is peculiar. A point can be assigned as noise even if it is positioned in between points which are assigned to a cluster. This phenomenon can be seen in Figure 5.5. The point second from the left in the OPTICS output is assigned as noise while the point farthest to the left is assigned to the cluster. This is due to the implementation of the noise assignment in this thesis. The threshold selected by the internal evaluation to produce the results in Figure 5.5 was 30 m. The reachability plot of the cluster together with the threshold can be seen in Figure 5.6. The second point from the left must be the first point of the clustering order and the point most far to the left must be the second point in the clustering order. Due to that the reachability-distance of the leftmost point is higher than the selected threshold, the prior point in the clustering ordering (the point second from the left) is assigned as noise.

Figure 5.5: To the left is a cluster taken from the output of the clustering procedure when OPTICS is used as the refining algorithm. To the right is the same cluster but when using DBSCAN as the refining algorithm.

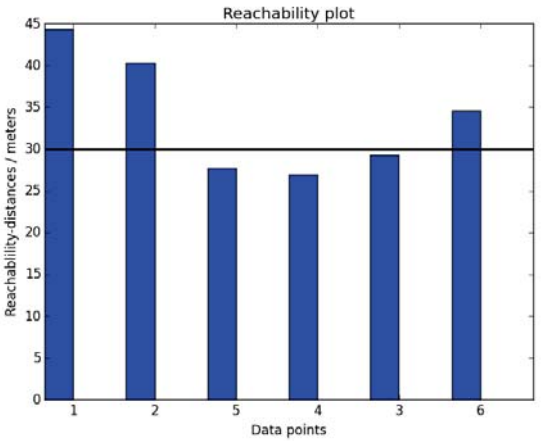

Figure 5.6: Reachability plot of the GPS points seen in Figure 5.5.

5.5.2 MSTC

As can be seen when comparing tables 5.1 to 5.6, the MSTC-algorithm has the highest separations value in all cases but two, where OPTICS is higher. This tells us that MSTC will separate points into separate clusters in areas which have been defined as a single cluster in the golden standard. This behaviour is a consequence of that the algorithm considers edges of the MST in order of increasing weights and that the merging function is dependant on the local density. If the area of data

points is multimodal, and the distances between modes are larger than the chosen threshold, the points will we separated in different clusters. This behaviour was hoped to be an advantage over DBSCAN but turned out to be a disadvantage with respect to the external evaluation. Figure 5.7 illustrates how MSTC will tend to separate data into clusters when it is not desirable and how DBSCAN will not.

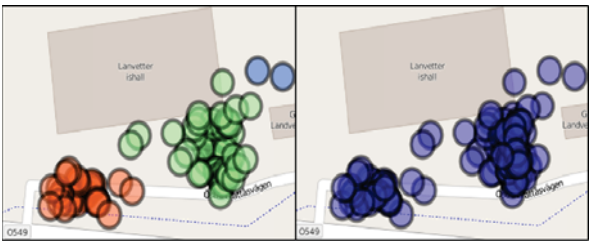

Figure 5.7: Caveat of MSTC. To the left is a cluster taken from the output of the clustering procedure when MSTC is used as the refining algorithm and euclidean distance is used as similarity measure. To the right is the same when using DBSCAN. Different colors represent different clusters and the blue points in the left image are noise.

5.5.3 Driving time

The driving time generated from the OSRM server may differ a lot from the actual driving time. As can be seen in Figure 5.8 the route between the data points generated by OSRM is over 300 meters wrong compared to the actual driving distance between the points. This is due to that when calculating the driving time, the points are first assigned to the nearest road, and then the distance between the nearest point on the nearest road for each original point is calculated. Too often, the nearest road is not the actual road from which the vehicle came from.

Figure 5.8: A shortest path generated by OSRM.

5.5.4 Silhouette Coefficient

The internal evaluation method has great influence on the resulting clustering. The silhouette coefficient, as mentioned, values cluster compactness and inter-cluster separation, and the way it calculates this value is by using average distance to all points in the same cluster and average distance to all points in the nearest cluster. This works best for clusters with spherical shapes. Also the silhouette coefficient does not make much sense for noise as there is no cluster compactness for a single point. This is handled by setting $a(i) = 0$, as described in section 2.4.1.1, which results in $s(i) = 1$, which is the best possible value for a point i. Hence, the silhouette coefficient tends to choose clusterings containing noise. A third problem is that the silhouette coefficient assumes that there are multiple clusters in the data set, however, using the developed procedure, there will be subsets of data which contain only one destination or perhaps none. If there is no nearest cluster in the subset, i.e. less than two clusters, then $s(i) = 0$. Hence a single cluster (and no noise) in a subset will not be chosen by the internal measure if an alternative is possible. A negative value of the silhouette coefficient can not occur using the clustering procedure as the resulting clusters will always have a (positive) distance between them. Figure 5.9 shows an example of when the aforementioned caveats of the silhouette coefficient take effect. It shows a subset of data given by the pre-clustering where the points are later wrongly classified as noise by the refining algorithm. This is due to that the lowest threshold is low enough to not cluster the three closest points together.

Figure 5.9: A subset of the data given by the pre-clustering where all points are incorrectly classified as noise by the refining clustering.

5.6 Adaptive procedure compared to DBSCAN

Table 5.10 shows the total cost for both the developed adaptive procedure (using euclidean distance and DBSCAN) and the original DBSCAN with $minPts = 3$ and $eps = 50m$.

Vehicle	T.C. adaptive ϵ	T.C. $\epsilon = 50m$
1	0.4322	0.1121
2	0.1493	0.1685
3	0.0174	0.1165
4	0.0254	0.0995
5	0.1810	0.1098
6	0.2104	0.1572

Table 5.10: Total cost of clustering results from the clustering procedure using Euclidean distance and DBSCAN, and clustering results from a simple DBSCAN with $\epsilon = 50$.

These values indicate that our developed procedure with adaptive parameters does not perform much better than DBSCAN without adaptive parameters. However, one must take into account that the parameter setting $eps = 50$ in DBSCAN was an educated choice, and all data sets evaluated have been from the Gothenburg area. This parameter value may not be as good of a choice in for example USA or China where road grids have other characteristics. Moreover, using the output of the external evaluation, we have identified that the main problems that the adaptive procedure has which DBSCAN does not have as often, are the problems with the internal evaluation method described in the previous section.

6
Conclusion

A clustering procedure which dynamically sets the algorithm parameter for different spatial regions using an internal evaluation measure has been developed. It has been tested and evaluated with different density based algorithms and two different similarity measures. The clustering procedure gives similar results independent of the choice of refining algorithm and similarity measure. It finds most of the meaningful destinations using any setup and the cost function penalties of the different refining algorithms do not differ significantly. This most probably has to do with the internal evaluation method. There is a clustering for each subset which will generate the highest silhouette score. This clustering or a similar clustering is often achieved by all the different algorithms. Hence the final clustering chosen for each algorithm will be similar to the final clustering chosen for the others. Therefore we think that to improve the clustering procedure, alternative internal evaluation methods should be tested or developed before considering to change the refining algorithm.
At the moment the similarity measure giving the best result is the euclidean distance. This is most likely due to the erroneous driving times generated by OSRM. If OSRM were to use a more detailed road map or handle points which are off the road grid differently the generated driving times could be correct, leading to better clustering results. Using distance between points as a criteria for clustering points together is effective in identifying separate destinations, but there are cases where it is not sufficient. In those cases, more information which can help in recognizing if the points are on a parking lot is needed.
The MSTC finds the most identified meaningful places but it also has a tendency to split up points belonging to the same destination. This would lead to skewed probabilities in an application where the data corresponding to specific destinations is used for predicting the next destination. OPTICS implementation performs the worst overall and has no outstanding properties. DBSCAN has a tendency to cluster points belonging to different destinations together but has the best overall results. To prove weather the developed clustering procedure is better than DBSCAN for clustering destinations, more data sets must be clustered and evaluated. Considering the adaptive procedure reached similar results of DBSCAN when the parameter of DBSCAN was manually set to perform well, the procedure shows promise. If the procedure can be modified to handle noise within the subsets better and subsets with less than two clusters, it will have a clearer advantage over DBSCAN.
We consider the goals listed in the Introduction to be reached. We have expanded on algorithms to develop a clustering procedure which adaptively sets the parameters based on local distribution of points. The relationship between the data points with respect to their position in the road network has been examined and used as a

similarity measure. We have also evaluated the results of different configurations of the clustering procedure.

Bibliography

[1] Gower, J., Ross, G. (1969) Minimum spanning trees and single linkage cluster analysis. Applied statistics.

[2] Felzenszwalb P., Huttenlocher D. (September 2004) Efficient Graph-Based Image Segmentation. IJCV 59(2)

[3] Ankerst M., Breunig M.M., Kriegel H.P., Sander J. (2008) OPTICS: Ordering Points To Identify the Clustering Structure. ACM SIGMOD international conference on Management of data. ACM Press. pp. 49–60.

[4] Rousseeuw P.J. (November 1986) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20

[5] Ester M., Kriegel H.P., Sander J., Xu X. (August 1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)

[6] Manning C. D., Raghavan P., Schütze H. (April 2009) An Introduction to Information Retrieval. Cambridge University Press

[7] Moulavi D., Jaskowiak P.A., Campello R.J.G.B, Zimek A. (2014) Density-Based Clustering Validation. Proceedings of the 14th SIAM International Conference on Data Mining (SDM), Philadelphia, PA

[8] Saitta S., Raphael B., Smith I.F.C. (2014) A Bounded Index for Cluster Validity. Ecole Polytechnique Fédérale de Lausanne (EPFL)

[9] Cordeiro de Amorima R., S., Hennig C., (2015) Recovering the number of clusters in data sets with noise features using feature rescaling factors. Information Sciences 324 pp. 126-145

[10] Arbelaitz O., Gurrutxaga I., Muguerza J., Pérez J.M., Perona I. (2012) An extensive comparative study of cluster validity indices. Pattern Recognintion 46 pp. 243-256

[11] Guerra L., Robles V., Bielza C., Larrañaga P. (2012) A comparison of clustering quality indices using outliers and noise. Intelligent Data Analysis 16 pp. 703-715

[12] Robert Geisberger. (2008) Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. (Thesis). Institut für Theoretische Informatik Universität Karlsruhe.

[13] Amigó E., Gonzalo J., Artiles J., Verdejo M.F. (2009) A comparison of extrinsic clustering evaluation metrics based on formal constraints. Information Retrieval 12(4) pp. 461-486.

[14] de Souto M. C. P., Coelho A.L.V., Faceli K., Sakata T.C., Bonadia V., Costa I.G. (2012) A comparison of external clustering evaluation indices in the context of imbalanced data sets. Neural Networks (SBRN) 2012 Brazilian Symposium

[15] Panahandeh G., Åkerblom N. (2015) Clustering driving destiniations using a modified DBSCAN algorithm with locally-defined map-based thesholds. Book of Abstracts ECCOMAS Thematic Conference CM3

[16] MacQueen J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability 1. University of California Press. pp. 281–297.

[17] Chakrabarti S., Ester M., Fayyad U., Gehrke J., Han J., Morishita S., Piatetsky-Shapiro G., Wang W. (2006). Data Mining Curriculm: A Proposal (Version 1.0). Intensive Working Group of ACM SIGKDD Curriculum Committee.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER´S THESIS

Date of issue
December 2015

Document Number
ISRN LUTFD2/TFRT--5995--SE

Author(s)
Carl Levin (LTH)
Christopher Håkansson (CTH)

Supervisor
Niklas Åkerblom, Volvo Cars Corporation
Giacomo Como, Dept. of Automatic Control, Lund University, Sweden
Rolf Johansson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle
Clustering driver's destinations - using internal evaluation to adaptively set parameters

Abstract

With advanced navigation systems becoming ubiquitous in modern cars, the availability of detailed GPS data opens up new research areas in the fields of pattern analysis and data mining. By capturing the end-of-trip GPS points of each trip made by a driver, that driver's meaningful destinations could be identified. The knowledge of these destinations can be used for route prediction, which in turn can be used for optimizing the motor control to decrease emissions. It can also be used for developing functions for autonomous vehicles. In this thesis a way of extracting these meaningful destinations from GPS data using clustering algorithms has been developed and evaluated. The result is a clustering procedure consisting of 2 steps of clustering. First a pre-clustering to divide the data into subsets corresponding to smaller spatial areas. Then, a refining clustering step for which the parameter of the algorithm is adapted to each subset. Adaptively setting the parameter for each subset is done by testing a set of parameters and evaluating the results internally, with the Silhouette coefficient, and choosing the parameter giving the best evaluation score. The best performing configuration of our procedure, according to our external evaluation method, is in par with the performance of DBSCAN with a supervised choice of parameter setting. Further evaluation of data sets from different areas of the world are needed to draw strong conclusions of the developed procedures performance.

Keywords
GPS data, clustering analysis, DBSCAN, OPTICS, adaptive parameter, destinations, Silhouette coefficient, road distance.

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-44

Recipient's notes

Security classification