Utrecht University

Improving visualization of trajectories by dataset reduction and line simplification

Student: Mihai Borcan
Student #: 3443949

Game and Media Technology master program
Supervisor: Marc van Kreveld

Abstract

When visualizing a dataset of trajectories, whether it's hurricane paths, pedestrians or car movements, overlapping and very similar trajectories cause the visualization to become obstructed and features of the overall set can go unnoticed by analysts. This clutter characteristic is common in data representation but hasn't been consistently defined and quantified.

The current thesis approaches the definition and measurement of clutter in the context of representing spatial trajectories and presents two methods of lowering the clutter values to a set degree. The first method investigates removing trajectories from the dataset in such a way as to maximize the improvement in the clutter measure. The second one is an adaptation of the Imai-Iri algorithm for line simplification, geared towards minimizing the intersection points within the dataset, thus lowering clutter.

The two methods, applied on a real dataset of human movement trajectories, result in significant improvements of the clutter measure relative to the established goal. Both methods are interpreted with regard to human perception and a conclusion is drawn on which method is more efficient depending on the goal of the user; further improvements to the method are discussed in the future work section.

Contents

Table of figures

1. Introduction

1.1. Trajectory data

Movement is regarded as the change in physical position of an object with respect to a set reference system, which is in most spaces a two or three-dimensional representation of geographic space. A trajectory represents the path made by the moving object through space and time.

A trajectory is the representation of movement of an object or person through that geographical space. Typically, trajectories are recorded through a series of positions in time, so a trajectory can be viewed as a function that attaches a location in (geographical) space to each time moment. Moment-related characteristics are the position at a particular moment, speed, acceleration or direction at a point in space, accumulated distance, etc. Overall characteristics are the geometric shape, length of a trajectory, duration, etc.

With today's technologies that include GPS, Bluetooth or Radio Frequency Identification (RFID), it has become easier to capture movement data at large scales, tracking a large number of objects for long periods of time with high sampling rates, at low costs. This has resulted in enormous datasets of trajectories of vehicles, people or animals, used for analysis and visualization. One of the biggest challenges is appropriately handling these constantly increasing datasets of trajectories for efficient visualization in areas such as biology, transport, engineering or services. This will be the focus of the present thesis.

1.2. Problem statement

Visualization of large datasets of trajectories raises multiple questions that have to be dealt with. Firstly, how can we display a large number of trajectories in an uncluttered fashion while preserving the general properties of the dataset and keeping global paths and points of interest intact, such as trajectory hotspots and highways? One major issue is the size of the dataset, which can considerably affect performance. If we attempt to display all trajectories as lines on a map, there is no way of differentiating between different entities, or distinguish transportation highways or hubs where many trajectories meet. There is a need for analysis and processing of the data, while preserving the global aspect. This can be achieved with several methods, such as clustering, aggregating with density maps, trajectory reduction, line simplification or by using post processing.

Secondly, how can trajectories be differentiated in the analysis and processing of a dataset? When comparing trajectories, do we take into account just the geometrical shapes, or can we account for other attributes, such as the temporal aspect, speed or direction? This includes start and end times, duration of the trajectories, but also sampling rate, which can affect the similarity of two trajectories. These can be summarized by use of a measure adapted for sequences that vary in time or speed. We also need to set trajectories apart by the impact they have on the overall visualization by using its own characteristics, such as length, number of sampling points or number of intersections with other trajectories.

Thirdly, how do we measure an improvement of a visualization technique and how do we decide whether the display is good enough? We need a measure of visual clutter that reflects both the high density of trajectory objects and the number of interactions, i.e. intersections, between them. In order to improve the visualization, the solution must either minimize the clutter value or lower it under a certain threshold. This threshold can be determined either computationally or

experimentally, by use of human judgment. In the end, the visualization of a data set of trajectories must satisfy the requirements imposed at the start and help human operators better analyze the data set.

1.3. Approach

This thesis offers solutions to the problems described above through the development of two methods that are expected to produce a better combined result than if applied individually.

To start with, I use two datasets of trajectories; the first represents more than 6000 trajectories of 178 human subjects, performing everyday activities, over a period of 3 years, mostly in the Beijing area – for practical reasons, only a subset of the data will be used, contained within a specified region. The second dataset, of roughly 600 items, contains pedestrian trajectories in the city of Delft. An initial step is to process these datasets in order to be accessible for fast reading. The methods are applied to five subsets of different sizes, which should yield comparable results and demonstrate their generality.

The two improvement techniques that will be applied to each dataset are **incremental dataset reduction** and **contextual line simplification**. In order to quantify the effect of these two methods, I first define the measure for visual clutter for a dataset. This is expressed as:

$$C_{window} = L_{window} + c * N + c' * I$$

This measure is applied for a determined area, in our case a square window. $\boldsymbol{L_{window}}$ is the total length of trajectories within the given window, $\boldsymbol{N}$ is the number of different trajectories found within the window and $\boldsymbol{I}$ is the number of intersections between trajectories (self intersections are not considered). We use constants $\boldsymbol{c}$ and $\boldsymbol{c'}$ to scale the integers to the order of magnitude of the average trajectory length; both constants are set to the value of the window edge.

Since clutter is most often a local instead of a global characteristic, the entire visualization is split into a grid of square elements, which will be used to calculate the level of visual clutter locally. Each grid element represents its own window and it will be used to calculate the clutter independently. This results in a clutter matrix, where the matrix represents the grid and each element holds the clutter value for that grid element.

In order to calculate intersection points within a given window, I adapt the line segment intersection algorithm presented in de Berg *et al.* [47]. The result will be an initial clutter value for each grid element. The values of ***desired_clutter***, which are calculated for each grid element, are set to a fraction of the initial values, depending on the goal of the application. These two arrays of values are part of a global function called the clutter improvement function, which is:

$$f(grid) = \sum_{grid} min\left(\frac{desired\_clutter}{actual\_clutter}, \frac{actual\_clutter}{desired\_clutter}\right) * \sqrt{desired\_clutter}\ .$$

The function is the sum across the grid of the ratio between the calculated clutter in each grid cell and the desired clutter value in that cell. Each ratio is weighted by the value of the respective desired clutter. This ensures that the more cluttered grid cells have a greater influence over the (global) clutter improvement function.

This is the function that the two methods presented below will try to optimize. By using a grid of 10x10 elements, when the actual clutter reaches the desired values, the function evaluates to its maximum value:

$$f_{desired}(grid) = \sum_{grid} \frac{desired\_clutter}{desired\_clutter} * \sqrt{desired\_clutter} = \sum_{grid} \sqrt{desired\_clutter}$$

The first method for improving visual clutter is **incremental dataset reduction**. At each step, the algorithm analyzes the dataset and determines which one trajectory to be removed that best satisfies the criterion used. The clutter is updated within each grid element and iteration continues in the same way.

There are two ways to choose which trajectory is removed. The first one is by comparing all pairs of trajectories and choosing the one with the smallest average "distance" to every other trajectory. This suggests that the trajectory removed has the least impact on the overall visualization. The distance to be used will be dynamic time warping. The second option is determining the trajectory that contributes most to global clutter, by count of length and intersection points. The way the measure is designed, removing trajectories incrementally means the clutter improvement function will increase gradually until a maximum is reached. The algorithm records at which iteration the function achieves its maximum and stores the remaining trajectories as best result.

The second method is **contextual line simplification**, where I attempt to minimize the number of intersections of a trajectory with other trajectories by adapting the Imai-Iri method ([48], [49]). The algorithm stores the intersection points and the trajectories that intersect at that point. The global clutter function is calculated and updated after each step. At each iteration, the algorithm takes a trajectory and performs line simplification in the context of intersection points, trying to find all possible allowed shortcuts between points and choosing the path with the least intersections with other trajectories.

The problem can be formalized as: given a sequence of points $P_1, P_2, \ldots P_n$, and an error ε, the algorithm calculates an approximating polygonal chain $P_{i_1}, P_{i_2}, \ldots P_{i_k}$, such that:

a) $i_1 = 1$ and $i_k = n$,

b) for each point P_j such that $i_k \leq j \leq i_{k+1}$, the distance between the point P_j and the segment $\overline{P_{i_k} P_{i_{k+1}}}$ is at most ε.

The approximating polygonal chain $P_{i_1}, P_{i_2} \ldots P_{i_k}$ represents the shortest path in a graph where P_i are nodes and all allowed shortcuts between points represent edges. Each edge is given a cost determined by the number of intersections contained and the edge length. The goal is to minimize the total cost while traversing the graph from P_1 to P_n by taking into consideration all permitted shortcuts.

The two algorithms are independent. The thesis will determine which module or combination of modules will produce the best improvement of the global clutter function, and also debate on how this value coincides with human perception.

1.4. Outline

The following chapters are structured as follows: Chapter 2 contains a literature review into the subjects of visualization and trajectory processing and display, grouped into categories according to the focus of the papers. Some publications are more general and present different visualization techniques used elsewhere in the literature; others employ various methods that can be adapted for trajectories, like data mining, graph visualization or density maps.

The contribution that this thesis brings to trajectory visualization is detailed in Chapter 3 where I give an in-depth description of how the clutter measure is devised, including visual examples, and what the two algorithm modules consist of.

Next, Chapter 4 presents information about the datasets that are used in the experiments, as well as how it was preprocessed for easier use in the algorithms.

I then go on to present the results of applying the algorithms to the datasets in Chapter 5. The methods are applied to sets of different sizes in order to determine how the scale of the datasets influences efficiency.

Chapter 6 draws conclusions regarding how the devised clutter measure and optimization algorithms behave and if the improvement is significant enough to be noticeable by human perception.

Finally, the last chapter presents the algorithms limitations and potential improvements. I also briefly give some different directions in which future research can go regarding the improvement of visualizations of trajectories.

2. Literature review

In the scientific community, improving visualization of trajectories is relatively new; researchers are trying to come up with solutions to an array of issues that concern visualization, such as displaying large collections of trajectories in an uncluttered fashion, taking the time component into consideration, working with 3D trajectories, effective and scalable clustering of trajectories, etc.

I have researched existing publications that are related to trajectories and visualization of trajectories, as well as data mining for trajectories and graph visualization, each of which has techniques that can be adapted for efficient visualization of trajectory data. They are presented in this chapter, split into topical categories, with the mention that some papers address more than one topic.

2.1. Surveys of visualization techniques

Some scientific papers take a general approach to the problem of visualization. Buchin *et al.* [1] perform a literature survey of how movement data can be visualized. The papers listed in [1] address the problem of displaying large amounts of trajectories, caused by high density and visual clutter, problem solved either by data processing, using multiple views, showing trajectories non-cartographically, etc. The methods are split into categories based on the part of the data space that is visualized, either geographical, temporal or attribute space:

- Basic spatial visualization, which present the trajectories as lines on a map (2D or 3D)
- Spatio-temporal aggregation, taking the temporal aspect into account, by using kernel density for example.
- Non-spatial visualization, showing attributes such as speed, acceleration, direction, or time budget, type of object, etc.
- Visualizations linked to data mining methods, where the data is pre-processed in order to be more easily visualized (example: Self-Organizing Maps).

Real-life examples of trajectory data and how visualization is different for each one are presented by Andrienko [2]. Motivations for analyzing movement are: studying behaviors of animals, transportation management, detecting bottlenecks, improving layouts of buildings and services, etc. As a first step, trajectories can be aggregated by origin – destination or by movement directions, but clustering techniques consider the whole route. Methods for improving visualization that are presented include clustering, spatio(-temporal) aggregation, pattern extraction; these shorten processing times when handling large amounts of data or allow comparisons of groups of trajectories instead of looking at individual ones.

A different approach is taken by Long [3], which presents the idea that maybe a combination of web open-source visualization libraries is better for visualizing multi-variate trajectory data. The web visualization libraries presented are ArcGIS JavaScript API, Processing, Google Maps API, timemap.js and others.

2.2. Trajectory data analysis and mining

Outlier detection is a primary step in many data-mining applications. Lee *et al.* [5] adapt the popular method of outlier detection for trajectory data. They present a partition-and-detect framework for trajectory outlier detection, the novelty being the sub-trajectory approach.

When analyzing large amounts of data, a balance between human operation and automation is needed. Andrienko *et al.* [6] work towards a compromise between making it easy for human operators and handling large datasets algorithmically. They propose an analytical framework that combines interactive visual displays with computational methods, both for better human understanding and more efficient data processing. In the same fashion, Schrek *et al.* [7] approach the problem of cluster analysis by initially considering Self-Organizing Maps (or Kohonen maps) and improving on them with interactive visualization techniques, providing human supervision to an algorithm that otherwise may not comply with operator expectations and desires. Users are allowed to input preferences and use domain knowledge at various levels of detail.

All trajectory analysis and processing use measures for evaluating trajectory similarities. Pelekis *et al.* [8] provide a number of different similarity measures (some of which have been considered for this thesis), based on basic parameters, such as space and time, or derived attributes, like speed, acceleration, direction change.

One aspect of data analysis which is often overlooked is approached by Monreale *et al.* [4], which poses the novel question of privacy of individuals that are being tracked, especially for GPS-obtained trajectories. The proposed method supposedly achieves anonymity in a dataset of trajectories by transforming the GPS trajectory data based on spatial generalization and k-anonymity, a transformation that doesn't affect the quality of cluster analysis.

2.3. Graph visualization

The problem of graph visualization is useful for trajectory visualization in multiple ways. Intuitively, trajectories can be regarded as graphs where the sample points represent nodes and the paths are edges between the nodes. Also, visualizing a large graph is conceptually similar to visualizing a dataset of trajectories, with sample points of a trajectories representing nodes in a graph and segments between sample points being edges of the graph. This analogy is used in section 3.3. Van Ham *et al.* [9] address the problem of graphs with a large number of nodes, where other graph drawing algorithms fail even for a few hundred nodes, by selecting a subset of the edges to create a skeleton of the graph, showing its intrinsic clustering structure. Then the structure is used to create an embedding of the graph, and the remaining edges are added to the image.

Edge bundling in graph visualization is another issue. Holten *et al.* [10] investigates decreasing visual clutter and revealing edge patterns by presenting a novel approach of self-organizing edges into bundles, without the use of a hierarchy. The result shows, according to the authors, significant clutter reduction and clearly visible patterns. By comparison, Telea *et al.* [11] combines edge bundling with simplified graph visualization by use of hierarchical edge clustering, and then rendering clusters at user-defined levels of detail with novel image-based techniques that improve the end result, such as shading, fiddling with luminance, saturation, hue, brushing and a new type of semantic lens for interaction. These tricks can be applied for better visualization in the case of trajectories, as well. Bagatelj *et al.* [12] contains a new technique for clustering as a solution to visualizing large networks; the purpose of this clustering method is to produce both intra-cluster graphs and inter-cluster graphs, suitable for adaptation to different representation conventions.

2.4. Density maps

Density maps can be defined as images that show an aggregate overview of large amounts of data. They are composed by combining multiple density fields into a single image. A density field represents the aggregation of a relevantly chosen subset of trajectories using one or several

attributes. After composing various density fields for subsets of trajectories chosen with a uni- or multivariate filter, relations between attributes may be revealed in the density map, like for example correlation between time of day and type of vessel (in the maritime domain), or between geographical positioning and aircraft company (in air travel).

There is a relatively large number of papers on the subject of using density maps to visualize trajectories, in particular for vessels. In Scheepens [14], trajectories are convolved with a kernel onto a density map, for better use of analysis by maritime operators. Visualization of the density map reveals hotspots for vessel paths in the form of a shaded height field. The specific improvements in this paper are increased computational speed by use of modern graphics hardware and several new visualization methods that bring out relations between subsets of data otherwise not visible. Scheepens *et al*. [15] present a method to explore multivariate trajectories using density maps, the novelty being that the density maps are used to visualize trajectories with multiple attributes. Another contribution is the addition of high-end graphic hardware that supports intense computational needs. The algorithm is applied to maritime use cases, revealing temporal patterns, unusual trajectories of vessels, ambiguity solving and risk assessment. In continuation of this research, Scheepens *et al.* [13] use expressions that allow the analyst to model domain knowledge. Expressions occurring in visualization are expressions for filtering, such as queries, expressions that compute derived data, or expressions that model the rendering. Willems [17] tries to approach the following research question: "How can analysts be supported to understand large amounts of trajectories with multiple attributes by using interactive, visual representations?" The techniques developed here are used for trajectories with multiple attributes and their efficient visualization, by direct depiction, summarization or pattern extraction.

Considering the clustering problem for trajectories, D'Auria *et al.* [16] adapts a density-based clustering algorithm using a simple distance between trajectories. The paper also contributes towards focusing on the temporal aspect of the trajectory data, to improve the quality of clustering.

2.5. Volumetric approaches

There is literature that specifically tackles the problem of (too) large datasets and how they can be dealt with. Andrienko [18] introduces traffic-oriented view and trajectory-oriented view. The traffic-oriented view is a time-ordered sequence of *traffic situations*, which represent snapshots of all the entities' positions and attributes at a specific moment in time. The trajectory-oriented view is just a set of all trajectories, as polylines describing the movement during the time period given. The difference is noticeable when applying different methods of aggregation and summarization, since the first view only handles points, which are much simpler objects than trajectories. Albuquerque *et al.* [20] also treats specifically the problem of high-dimensional data. For this there is a need for automatic reduction techniques, automatic sorting or selecting the projections. Applying quality measures acts as a sort of filtering for visualization. This paper proposes a number of quality measures for a few visualization methods. Andrienko *et al.* [22] gives an approach to extracting meaningful clusters from large databases of trajectories by combining clustering and classification, while being assisted by a human operator through an interactive interface. The algorithm starts with an appropriately chosen subset of the trajectories and clustering is applied to it. The human operator can tweak the clustering parameters with respect to the analysis goals. Then a classifier is built, which is used to attach new objects to the existing clusters based on a measure of confidence. Each trajectory is either attached to a cluster or remains unclassified.

There are other methods to meaningfully display large collections of trajectories. Wang *et al.* [21] gives an adaptation of an interactive visualization system, applied for efficient viewing of all the

available hurricane trajectory data in the past 160 years. There are three linked views: spatial view, showing geometrical trajectories; temporal view, showing patterns in time; parallel coordinates plot showing multiple properties of the multi-dimensional data. This is a less-used approach for the handling of large datasets with multivariate data and its visualization. Other techniques are found in Schirski *et al.* [19], for flow visualization of particles in the form of trajectories. The same ideas can in principle be applied to potential 3D trajectory visualizations. The paper stresses the visual component, using a combination of suitable textures and illumination to create, in fact, an illusion of three dimensional tubes.

2.6. Visualization systems

Some systems bring one or more specific improvements to visualization, usually for a specific purpose. Guo *et al.* [28] is the inspiration for the hurricane visualization in [21]. It presents a visual analytics system for exploring and analyzing complex traffic trajectories. It possesses the same three perspectives (spatial, temporal and multidimensional). A particular application is detecting and analyzing microscopic traffic patterns and abnormalities. Similarly using multiple views, Lampe *et al.* [29] explores the idea of interactive difference views, enabling temporal trend discovery and analyzing large amounts of data using frequency-based visualization based on kernel density estimates. Users can compare different categorical attributes (ship type, wind direction) or quantitative attributes (how two hours' traffic compares to the average).

Hurter *et al.* [25] shows a complex system that tries to simplify visual analysis for aircraft trajectory visualization. The software, called FromDaDy, is based on scatter plots, brushing, juxtaposed views and other visual enhancements. By visual interaction (brushing or pick and drop) the users are able to define complex visual queries for data subset selection. Another similar enhancement can be found in Hurter *et al.* [26], which "proposes a semantic lens which selects a specific spatial and attribute-related data range. The lens keeps the selected data in focus unchanged and continuously deforms the data out of the selection range in order to maintain the context around the focus". The method is easily applicable to visualization of trajectory data, if the desired purpose is to have an overall view of the data while drilling more deeply for specific regions. Yan *et al.* [24] deals with semantic representations of trajectories. The paper presents a system that exploits 3rd party information sources to semantically enrich trajectories. The construction module computes and annotates all meaningful parts of heterogenous trajectories, while the visualization interface works on different levels of detail, from low-level GPS data (raw) to high-level semantic trajectories (viewing the trajectories as a series of relevant events in space and time).

A novel approach using probabilities, Wang *et al.* [27] shows a framework that uses a nonparametric Bayesian model, used for unsupervised trajectory analysis and semantic region modeling, with application in surveillance. The novelty (compared to other methods) is that trajectories are treated as documents and observations of a trajectory (such as points) are words in that document. Trajectories are clustered according to their likelihood. Words are clustered into topics (whose number is automatically decided) and the documents/trajectories that contain these objects are clustered together.

2.7. Domain-specific visualizations

Some papers have specific domains of applicability, such as vessel trajectories and traffic information, which means that the novel approach is developed with this specific application in mind. Willems *et al.* [30] is a user study that tests the performance of three types of visualization techniques with respect to three movement features that occur in vessel trajectories; more

specifically, the paper compares the density method presented in [13] with other well-known trajectory visualization techniques: animation of moving dots and the space-time cube. Furthermore, the tasks involved in the analysis process are specific to maritime data. The best performance is obtained for finding stopping objects, while comparing lanes and detecting fast movers performs at the same level as other methods. Grundy *et al.* [31] is another application-centric paper that contains some novel visualization techniques. The goal here is to analyze and find new patterns of animal behavior. Visualizations include spherical scatter plots, spherical histograms, feature-based state diagrams, etc.

3. Algorithm

3.1. Clutter definition and measure

Visual clutter on maps has been investigated for a long time, as far as Philips [33], which researches the idea of improving map readability by removing topographic symbols. The experiments carried out on a 1:50000 geological map revealed that point symbols clutter other points and line symbols obstruct other lines; removal of these symbols improved map readability in most cases. A more precise definition is given by Peng *et al.* [34], according to which visual clutter is "any aspect of the visualization that interferes with the viewer's understanding of the data" by obstructing patterns and structure. However, in [33] and [34] there is a contradiction between what is apparently clutter and what effectively impairs viewer's analysis of the visualization. Experiments carried out by Philips [33] reveal that the obstruction of map symbols doesn't depend only on the amount of clutter, but also the form taken.

There are many other understandings of visual clutter, focusing on which map elements obstruct the understanding and extraction of patterns, either quantitatively or qualitatively. For this thesis, I chose to loosely define clutter as the totality of features of the map that negatively affect readability and pattern analysis. Similarly, the reduction of visual clutter is defined as the process of removing these features in order to improve readability. The terms "clutter" and "clutter improvement" need to be quantified, so that we can differentiate between levels of clutter.

It is essential to precisely establish a measure of clutter that quantifies how obstructed a visualization is. The importance of a reliable measure of clutter is presented in Rosenholtz *et al.* [36], where they restrict the definition of clutter as being "the state in which excess items, or their representation or organization, lead to a degradation of performance at some task". This highlights the facts that clutter is not merely visual, but it is associated with performance degradation, and that clutter is also task-specific. The paper introduces The Statistical Saliency Model, which detects unusual items based on contrast, color, orientation or motion. A feature is more unusual, or salient, if its characteristics are "outliers to the local distribution of features in the display". Using this model, the authors introduce the Feature Congestion measure which locally measures the size of a covariance ellipsoid of the feature distributions and compares it to the available display.

While developing a measure for visual clutter, Janssen and van Kreveld [35] list several elements that influence the clutter in a map: point features (with size and color), line features (length, color, thickness), polygon features (size, color and texture) and boundaries (length and contrast). Their measure calculates clutter in the vicinity of any point and takes into account points, lines and areas around the given point. One limitation of their formula is that it doesn't consider interactions between elements as being a part of visual clutter, just independent features. The paper goes on to split a vector-based map into a grid of elements of k by k pixels and calculating the clutter value for each one, creating a clutter matrix.

This thesis adapts the methods used in [35] and improves on them by taking into consideration interactions between trajectories. The entire display of trajectories is superimposed onto a grid of 10x10 squares. The side length is determined by auto fitting all the trajectory elements within the grid.

When deciding which elements contribute towards visual clutter, not only the number and length, but also interactions between trajectories are considered. Firstly and the most evident contribution to clutter is the number of different trajectories, meaning that more trajectories imply a more cluttered display:

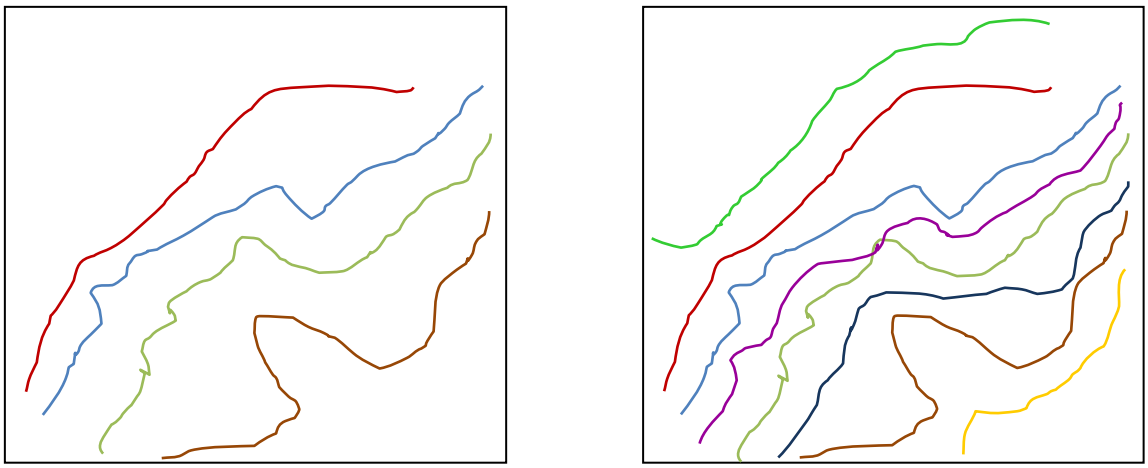


Figure 1 - Two examples of how the number of trajectories raises clutter. The second image contains twice as many trajectories of random lengths compared to the first one.

There is the debate whether a single trajectory and two halves of that same trajectory contribute the same towards clutter. This is not true for two reasons. Different trajectories are always represented with different colors, so one can tell them apart, whereas one trajectory is uni-colored. Furthermore, when it comes to processing, handling two trajectories takes double the number of iterations compared to the one trajectory.

Secondly, the length of a trajectory is directly proportional to the impact it has on the overall clutter values. In the following, the red trajectory has a greater influence on the clutter value than the blue one, being 60% longer.

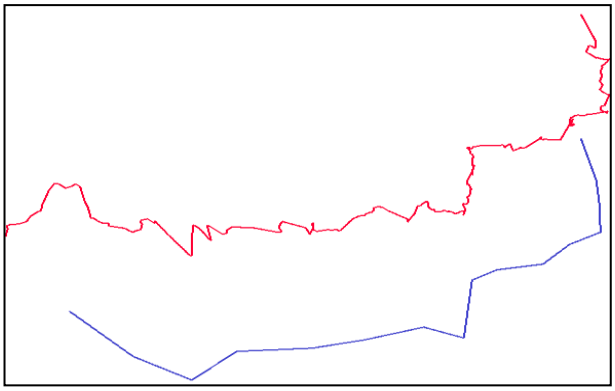


Figure 2 - Example of the influence of trajectory length over clutter

Thirdly, the interactions between trajectories have an impact on how cluttered a map is. Thus, line intersections have to be considered in the measure of clutter.

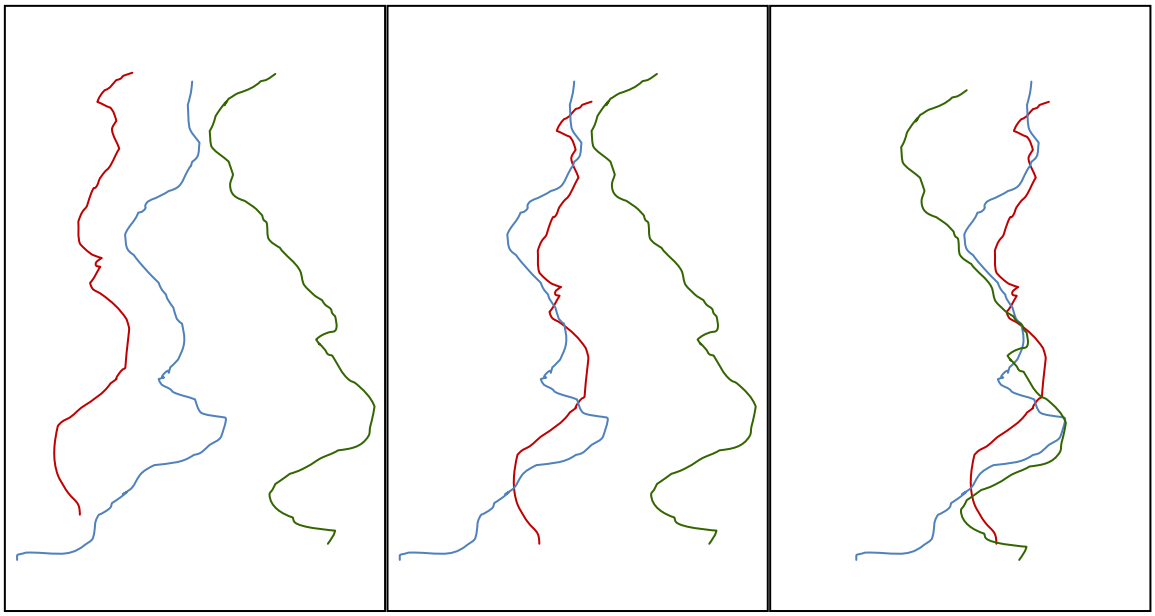
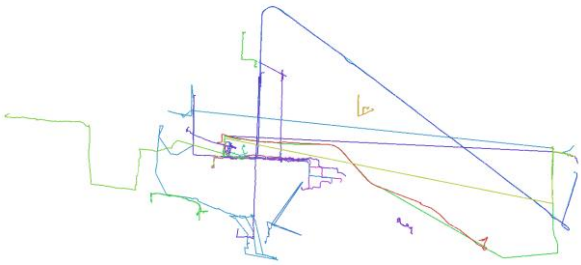


Figure 3 - Examples of how intersections between trajectories influence clutter; a) three trajectories without interaction; b) and c) two trajectories intersect, and d) three trajectories intersect each other.

The three elements that contribute to clutter that are taken into consideration are the number of trajectories, their lengths and the number of intersection points between different lines, as depicted above. The clutter measure chosen, which includes all three, is:

$$C_{window} = L_{window} + c \cdot N + c^{'} \cdot I$$

where L is the length of all trajectories within a given window, N is the number of different trajectories and I is the number of intersection points. The formula also needs constants c and c' to scale N and I to the magnitude of the trajectory lengths. The clutter measure is absolute, meaning that zooming in does not influence the value of the clutter over the initial window. Figure 4 illustrates a few examples from the data that demonstrate the suitability of the formula above.



(a) Number of trajectories:25
Total length: 1.96.
Intersections: 520
C = 102.42

(b) Number of trajectories:50
Total length: 3.40.
Intersections: 1924
C = 367.27

(c) Number of trajectories:100
Total length: 7.24.
Intersections: 8115
C = 1521.53

(d) Number of trajectories:200
Total length: 14.13.
Intersections: 34495
C = 6409.52

Figure 4 - Four datasets and the clutter values they create

I introduce a global function named the clutter improvement function, which is relative to the desired levels of clutter that the algorithm strives to achieve. This function takes the clutter matrix at a particular algorithm iteration as input, as well as the desired clutter matrix (determined at algorithm start, which remains constant) and outputs a value.

$$f(grid) = \sum_{grid} min\left(\frac{desired\_clutter}{actual\_clutter}, \frac{actual\_clutter}{desired\_clutter}\right) \cdot \sqrt{desired\_clutter}$$

The function is equal to the sum (over the entire grid) of the ratio between the current (actual) clutter in each grid cell and the desired clutter value for that cell. If the ratio exceeds the ideal value of 1, it is inverted; for the algorithm, an over-cluttered cell is equally bad as an over-simplified one. Each ratio is weighted by the value of the respective desired clutter. This ensures that the more cluttered grid cells have a greater influence over the (global) clutter improvement function.

If the desired level of clutter in each grid element is, for example, ¼ of the initial values, the function outputs in the beginning:

$$f_{initial}(grid) = \sum_{grid} \frac{\sqrt{desired\_clutter}}{4},$$

while the optimal value of *f*, assuming the target of ¼ of the clutter values is reached, is

$$f_{optimum}(grid) = \sum_{grid} \sqrt{desired\_clutter} = 4 \cdot f_{initial}(grid).$$

The goal of the algorithms is to maximize the value of $f(grid)$ towards $f_{optimum}$. The task of establishing what proportion of the existing clutter is optimal is heavily user- and application specific, and it is left to future researchers.

3.2. Incremental dataset reduction

The first of the two modules of the algorithm removes trajectories from the dataset incrementally in order to improve the global clutter function value. At every iteration, the algorithm considers eliminating one trajectory from the set whose removal improves the clutter matrix values and the clutter improvement function.

There are two criteria for determining which trajectory is to be removed, both having specific advantages. The first one is by comparing all pairs of trajectories and choosing the one with the smallest average "distance" to every other trajectory, where the distance is dynamic time warping ([40], [41]). This suggests that removing the trajectory has the least impact on the overall visualization, because it is the "closest" to all others. Otherwise put, given a set of trajectories $t_1, t_2, ...t_{n+1}$, the trajectory to be removed from the set is the one that minimizes the value $\sum_{i=1}^{n} DTW(t, t_i)/n$, where *DTW (t, t_i)* is the dynamic time warping distance between two sequences of points formed by trajectories *t* and t_i. After removing a trajectory, *n* decreases by 1 and the next trajectory to be removed is selected from the remaining dataset.

The second way of determining which trajectory will be removed is by calculating the contribution that each of the trajectories brings to the overall clutter. Given a measure $C_{traj} = L_{traj} + c \cdot I_{traj}$ for each trajectory, where L_{traj} is the length of the trajectory line and I_{traj} is the number of intersections that this trajectory has with all others, the trajectory with the greatest C_{traj} value is considered to contribute most to local and global clutter values and is to be removed.

After each removal, the clutter values are updated for all grid elements and the value of *f(C)* is computed. The algorithm records which iteration outputs the largest value for the clutter improvement function and stores the dataset at that iteration.

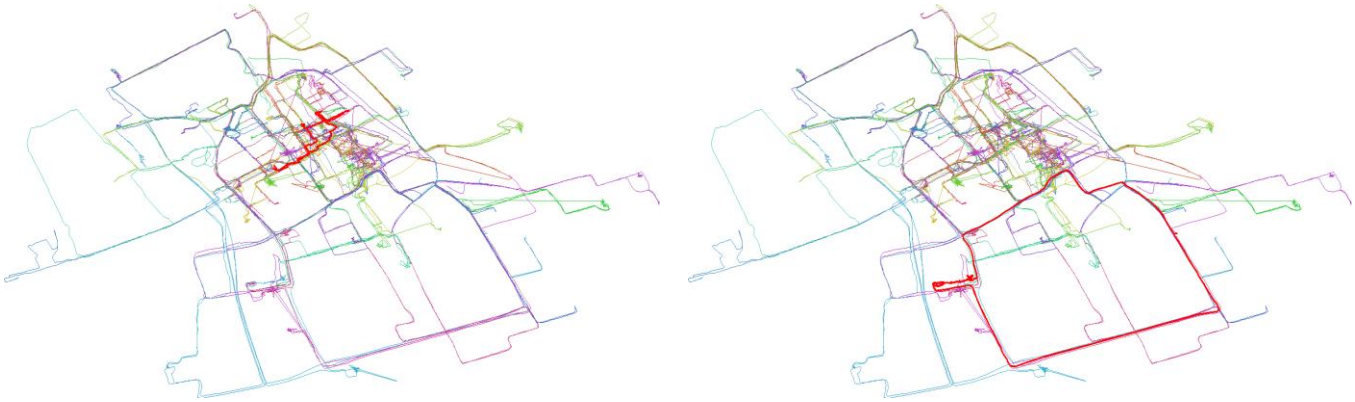


Figure 5 - Dataset reduction using the two methods. The best choice to be removed is shown with a thick red line. Left: dynamic time warping; right: the clutter contribution measure.

3.3. Contextual line simplification

The second module of the algorithm improves visualization by applying line simplification with regard to the number of intersection points on a particular line. The algorithm is adapted from the Imai-Iri polygonal curve approximation algorithm ([48]) as well as follow-up papers ([49], [50]).

Formally, let $P = (P_1, P_2, \ldots P_n)$ be a sequence of points that form a polygonal line, and an error ε. The contextual line simplification algorithm outputs a subsequence of points $(P_{i_1}, P_{i_2}, \ldots P_{i_k})$, with $i_1 < i_2 < \ldots < i_k$, that approximates the original line and that satisfies the following:

a) $i_1 = 1$ and $i_k = n$;

b) for each point P_j such that $i_k \leq j \leq i_{k+1}$, the distance between the point P_j and the segment $\overline{P_{i_k} P_{i_{k+1}}}$ is at most ε;

c) the approximating polygonal chain $(P_{i_1}, P_{i_2}, \ldots P_{i_k})$ represents the shortest path in a graph constructed from the original sequence of points P, where P_i are nodes and all edges in the graph are given a cost according to the number of intersection points it contains and the length of the edge: $Cost(edge) = I_{edge} + L_{edge} / L_{trajectory}$. The edge length is normalized by the total length of the original polygonal chain, so that in case of equal number of intersection points the algorithm still chooses the shortest distance.

Within the algorithm of line simplification, choosing the error ε becomes very important: a too high value will cause over-simplification of the line, while a value that's too small renders the algorithm inefficient, reducing too little intersection points. Figure 6 illustrates a trajectory with all possible shortcuts and their associated cost.

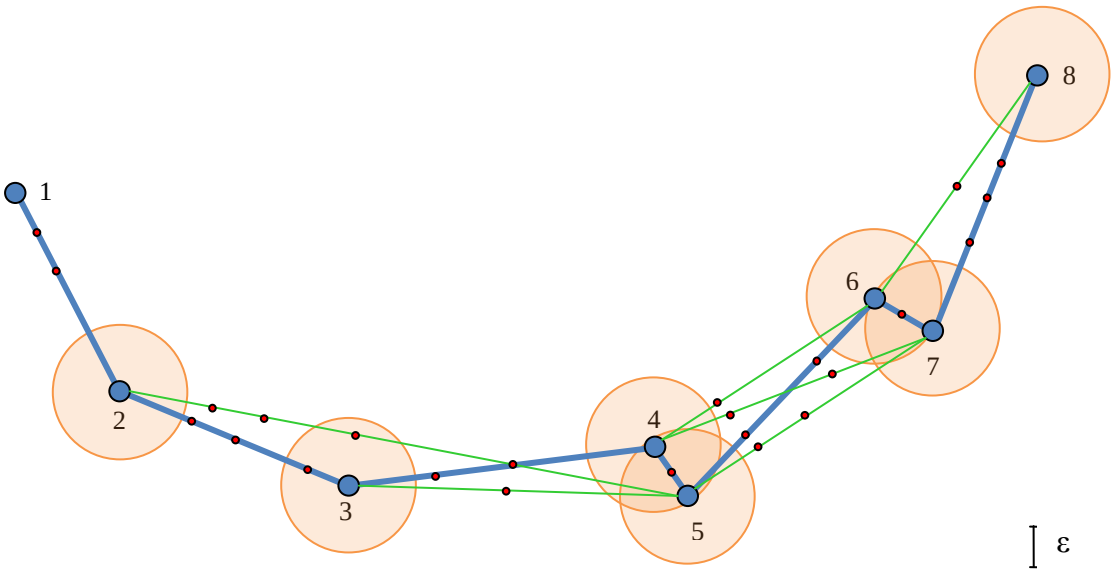


Figure 6 - Graph for line simplification. Green edges represent allowed shortcuts, intersection points are marked with red. Initial trajectory has cost 14, shortest path is 1-2-5-6-8 with cost 9.

4. Data sets

The experiments in this thesis make use of two data sets of trajectories. The first set comes from (Microsoft Research Asia) Geolife Project [43] and it consists of trajectories of 178 users, performing everyday activities as well as entertainment and sports, in a period of over four years (April 2007 - Oct. 2011). A trajectory in this dataset is represented by a sequence of time-stamped points, each of which contains the information of latitude, longitude and altitude. The entire dataset is made up of 17,621 trajectories with a total distance of roughly 1.2 million km and a total duration of 48,000+ hours. The trajectories have been recorded by different GPS devices, and have a variety of sampling rates, most of which are logged in a dense representation, meaning every 1 to 5 seconds or every 5 to 10 meters per point.

The files were initially organized by user in .PLT format, with each file containing a single trajectory. The 2D position is expressed by latitude and longitude in decimal degrees. The files contain other information, as altitude in feet, date and time. The trajectories have been combined into one .SHP file for faster accessibility, keeping only the geographical position. Each trajectory has a global ID and is represented by a MultiLineString object.

The second dataset has been procured through academic channels and it originates from iTravel Tech Inc (http://www.itravel-tech.com); it contains 636 trajectories of pedestrians in the city of Delft, The Netherlands. They come in .SHP format. The trajectories were originally stored individually as a succession of points, with other attributes such as altitude, date and time:

the_geom	LAT	LON	ALTITUDE	DATE	TIME	DATETIME
POINT (4.360540106380509 52.01549424988395)	+52,0154942	+4,3605401	11,4	28-4-2010	15:48:25	28-4-2010 ...
POINT (4.360280066198637 52.01534423220938)	+52,0153442	+4,3602801	12,1	28-4-2010	15:48:30	28-4-2010 ...
POINT (4.360019025802971 52.01518621352009)	+52,0151862	+4,3600190	13,1	28-4-2010	15:48:35	28-4-2010 ...
POINT (4.359735982164526 52.015038196140985)	+52,0150382	+4,3597360	13,9	28-4-2010	15:48:40	28-4-2010 ...
POINT (4.359421935008318 52.015065200291396)	+52,0150652	+4,3594219	13,0	28-4-2010	15:48:45	28-4-2010 ...
POINT (4.359201902299168 52.01513520943544)	+52,0151352	+4,3592019	12,1	28-4-2010	15:48:50	28-4-2010 ...
POINT (4.358913858483059 52.01506220154471)	+52,0150622	+4,3589139	10,3	28-4-2010	15:48:55	28-4-2010 ...

Figure 7 - Query of a single trajectory in the second dataset

A subset of these trajectories has been considered, being part of a concentrated, more dense area. Sample visualizations are given in Figures 9 and 10.

Figure 8 - Set of 500 trajectories from the first dataset

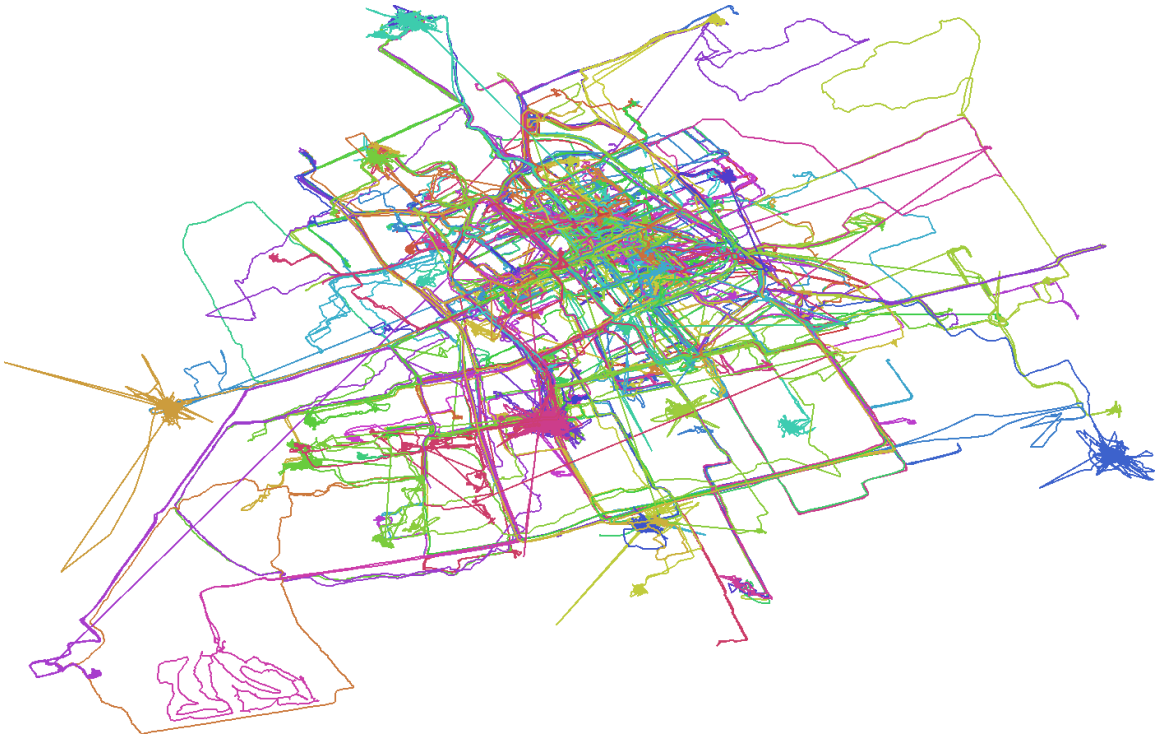


Figure 9 - Set of 500 trajectories from the second dataset

5. Experimental results

In order to determine the improvement of the visualization when applying the two techniques, experiments were carried out in groups on subsets of the original data. These subsets are composed of 100, 200 and 500 trajectories respectively. The purpose was to investigate if scaling up in data size influences the algorithm performance or efficiency in any way.

All datasets contain trajectories with up to 1000 sample points, since most trajectories that have thousands contain many redundant points. As can be seen in the following figures, the trajectories for this dataset follow some of the same popular paths.

Figure 10 - 100 trajectories from the first set

Figure 11 - 200 trajectories from the first set

Figure 12 - 500 trajectories from the first set

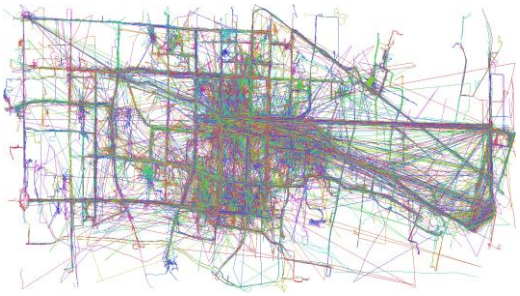


Figure 13 - full entire set of 6706 trajectories

Figure 14 - 100 trajectories from the second set

Figure 15 - 200 trajectories from the second set

5.1. Experimental set-up

For each dataset of trajectories, a grid of 10x10 square elements is superimposed on the visible frame of the set, in order to compute a matrix of clutter values. Initial calculations include determining which trajectories pass through each grid element, resulting in an array of collections of subtrajectories. In the end, each grid element will correspond to a collection of subtrajectories and a set of intersections of these subtrajectories, which gives all the information necessary to compute the clutter value within that element.

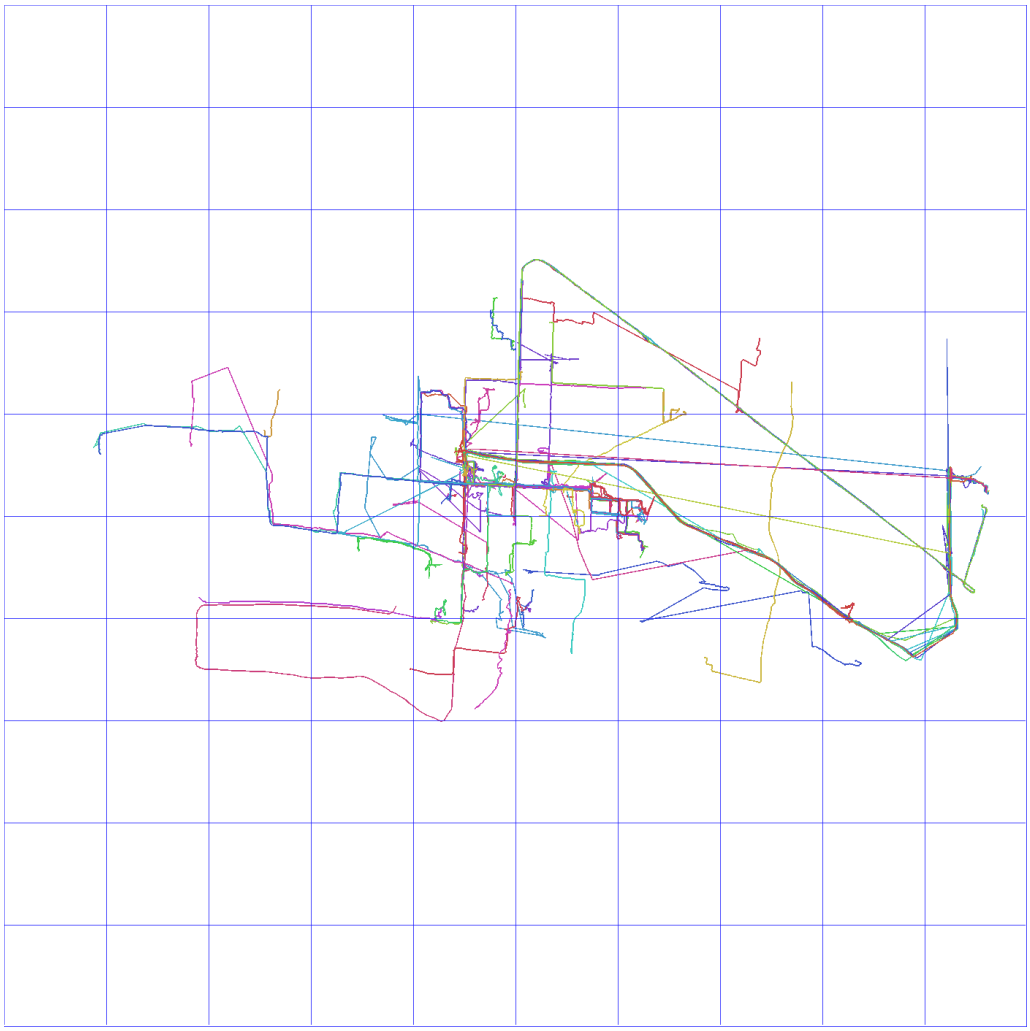


Figure 16 – Superimposing a grid on the dataset window

The matrix of computed clutter values also helps calculate the desired values, the ones that the algorithms will strive to achieve. For all main experiments, a goal of 50% desired clutter in each grid cell is set.

5.2. Small dataset

The dataset reduction algorithm is applied to the first subsets of trajectories (of size 100) using the two criteria mentioned. At each iteration, the trajectory to be removed is chosen either by the dynamic time warping or by the clutter contribution measure.

The evolution of the clutter improvement functions for the three methods are shown in the graph below, for comparison. Each algorithm run records the maximum value of the global function. For illustration purposes, all iterations are shown. The initial value of the clutter improvement function is 22.23, thus the desired value will be 44.46.

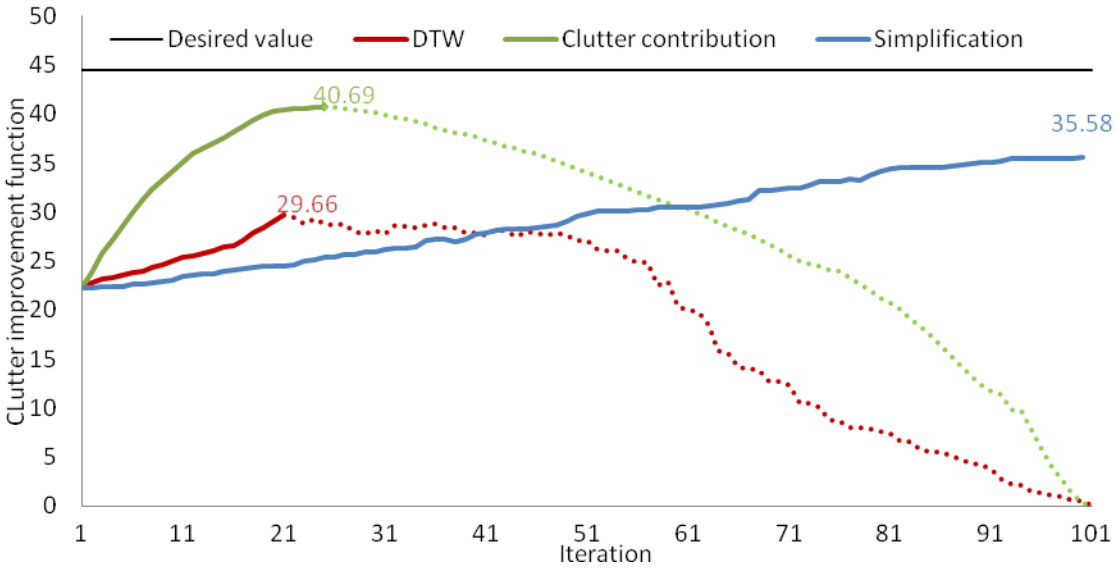


Figure 17 – Running the three algorithms on the first dataset. The desired global function value is 44.46

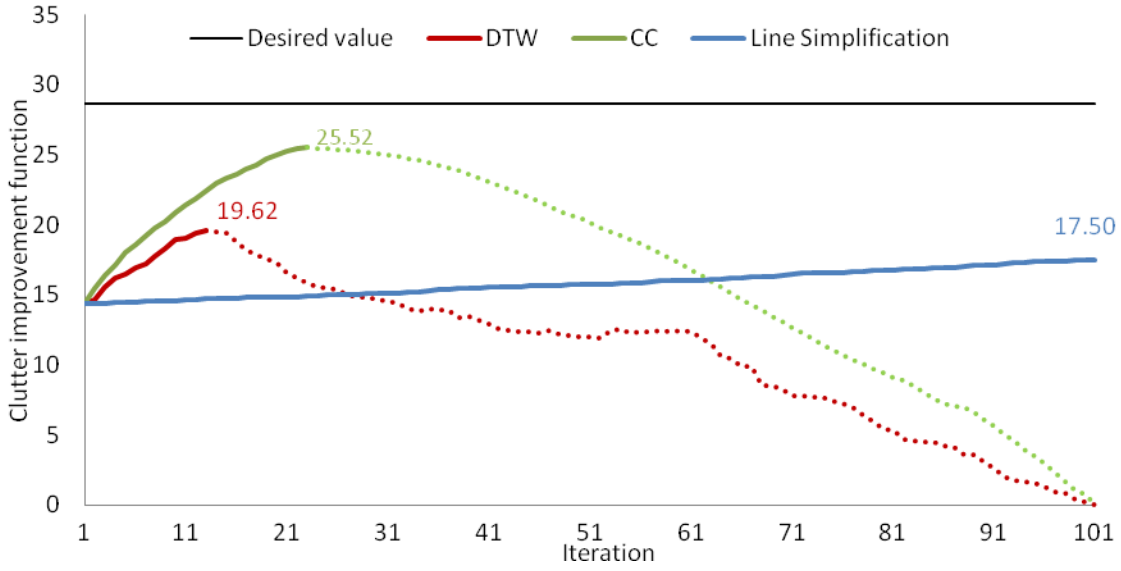


Figure 18 - Running the three algorithms on the second dataset of 100 trajectories. Initial value is 14.35, desired value is 28.70

The clutter contribution method performs better in the sense that it is able to achieve the maximum value of the three methods, without causing oversimplification in the dataset. It also approaches closer to the goal set by the algorithm, to achieve the desired clutter improvement function.

(a)

(b)

Figure 19 – Comparison of first original (a) and reduced (b) datasets using the dynamic time warping measure

(a)

(b)

Figure 20 - Comparison of first original (a) and reduced (b) datasets using clutter contribution measure

Above we can observe that the DTW method mainly removes trajectories that are situated closest to the center of the visualization, which means that the clutter measure in the most cluttered cells will get the greatest improvement. The clutter contribution method removes mainly trajectories that are longest and have the most intersections with other trajectories. This observation is supported by the figures below, which illustrate the values of clutter in the ten most crowded grid cells and the improvement percentage.

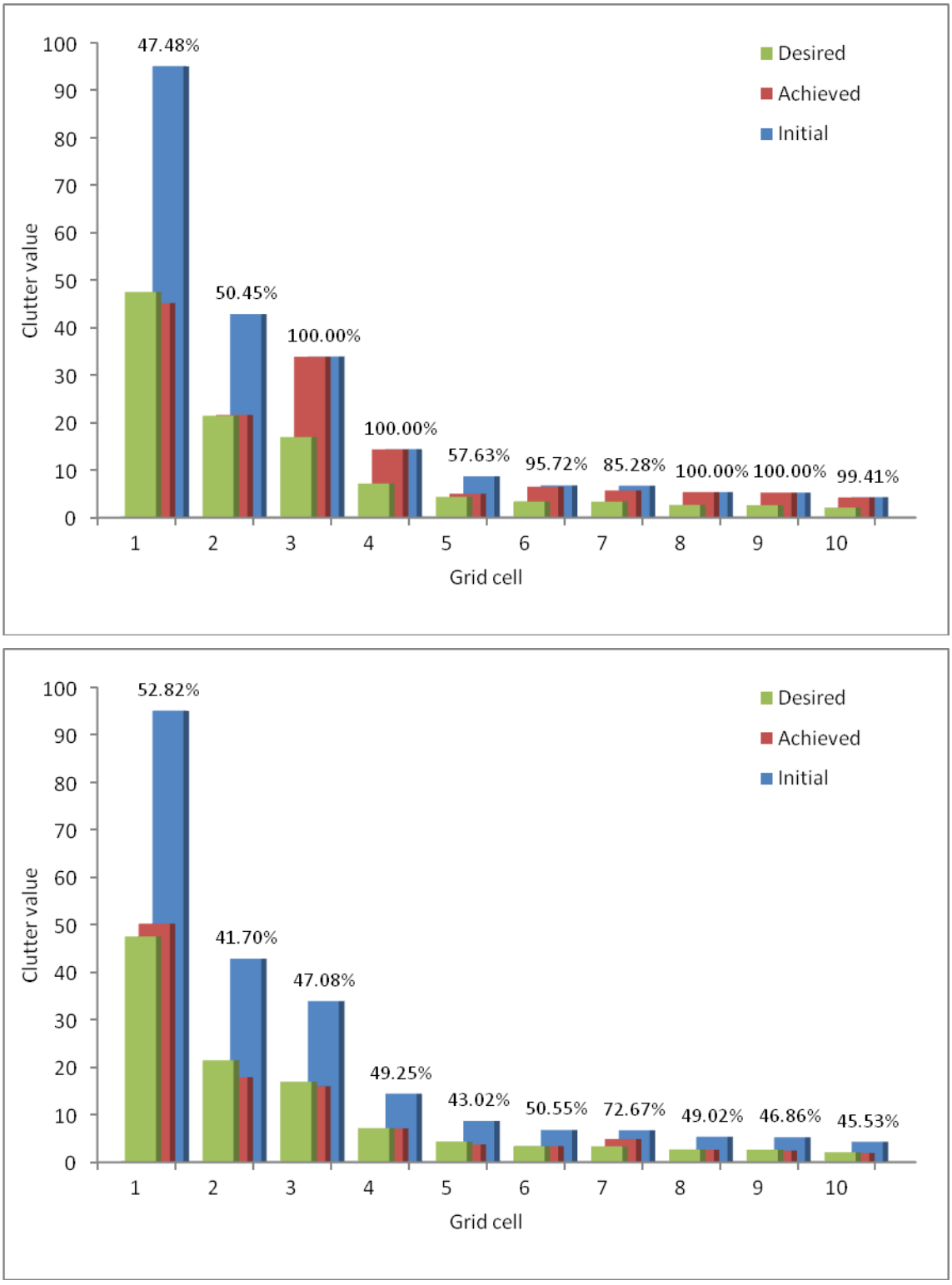


Figure 21 - Values for the ten most cluttered grid elements. The percentage shows the achieved proportion of clutter compared to the original. Above: DTW, below: clutter contribution

The line simplification algorithm applies simplification in the context of intersection points to all trajectories in the dataset. The ε constant is set to a value relative to the grid, which is 1/200th of a grid cell edge in the main experiments. It is possible to get a greater degree of simplification with

a greater ε (section 5.5). The line simplification has the advantage of keeping all trajectories and preserving their shape (if ε is small enough) and thus the two datasets appear to be very similar. It lowers clutter proportionally, but to a lesser degree than the clutter contribution method.

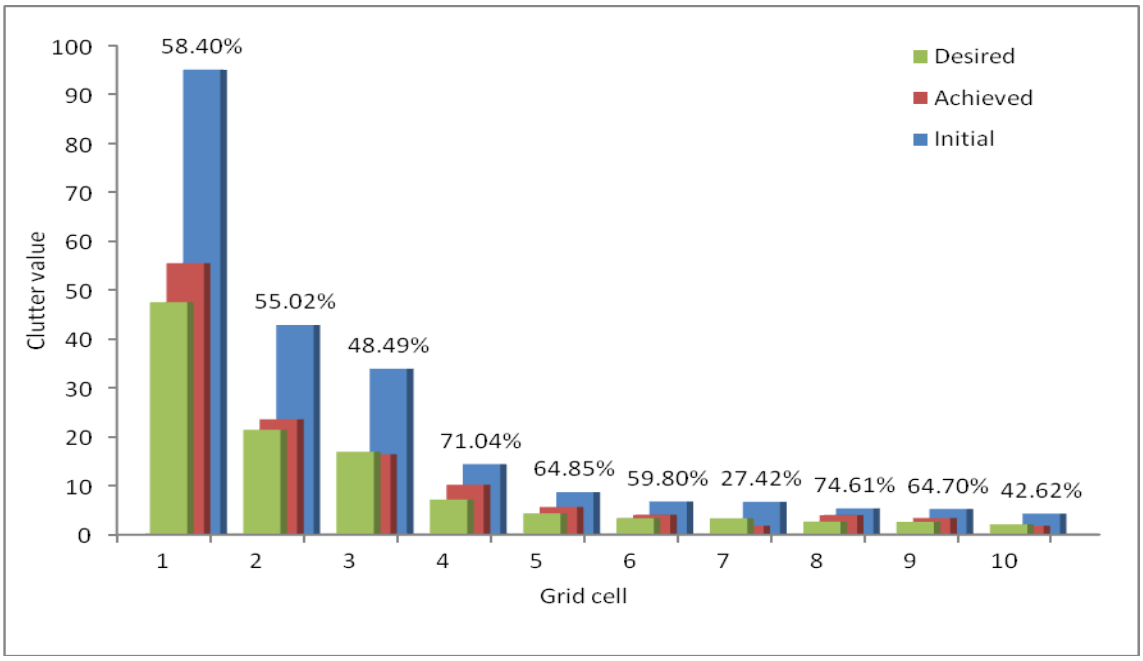


Figure 22 - Values for the ten most cluttered grid elements when applying line simplification

The results above indicate that minimizing intersection points is not enough and perhaps a combination of methods gives the best overall result. If applying simplification and reduction in succession, the first part removes many intersection points, lowering clutter to ~56% in the above example; dataset reduction is then applied, where number and length of trajectories carry a greater weight.

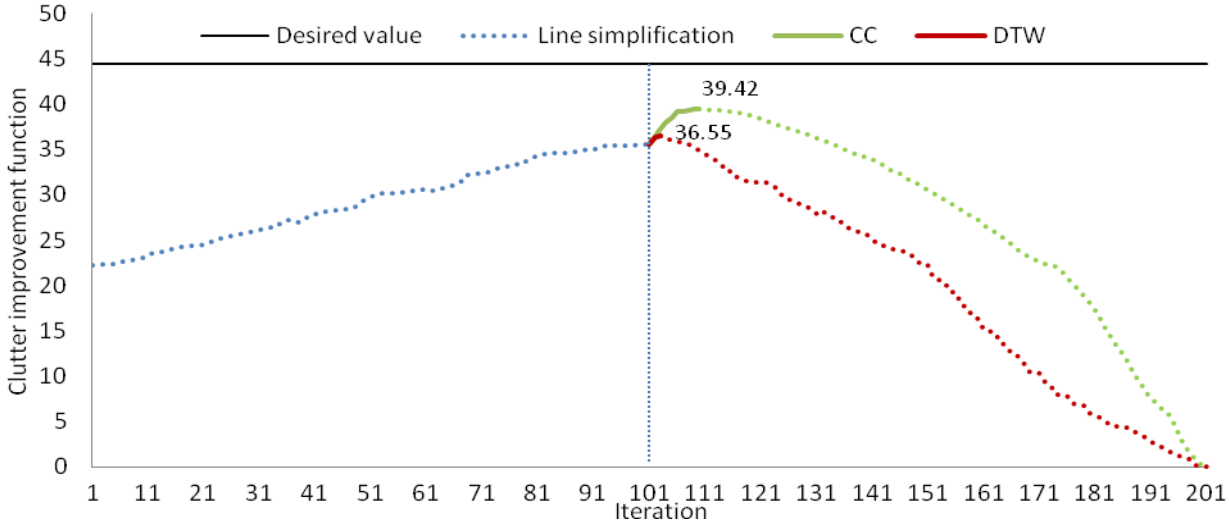


Figure 23 - Evolution of clutter improvement function when running line simplification algorithm and each variant of the dataset reduction algorithms. The peaks are marked with numeric labels.

Figure 24 shows how a combination of line simplification followed by dataset reduction is comparable to the best results obtained by applying reduction using the clutter contribution measure, but with far fewer trajectories removed from the dataset. Local performance is illustrated in figure 24.

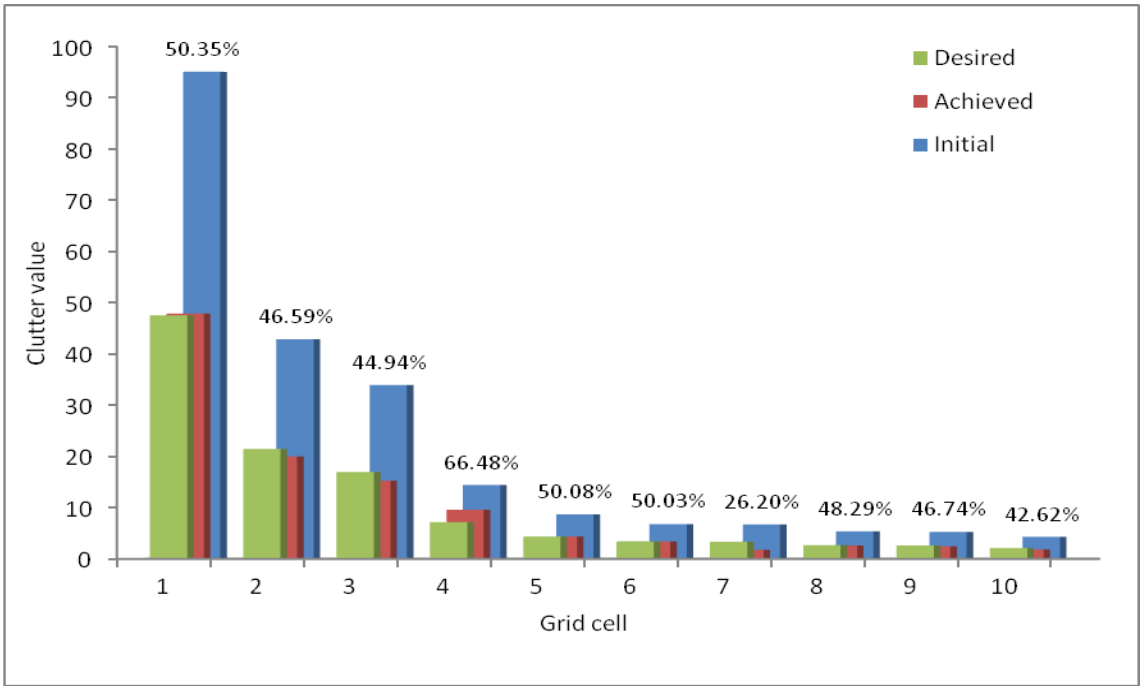


Figure 24 - Clutter values obtained when applying line simplification and dataset reduction CC

A combination of dataset reduction and simplification means the line simplification algorithm will be applied on a set where no more improvements can be made by removing trajectories. The choice is counter-intuitive and experiments show that it has the poorest performance of all combinations, resulting in over-simplification.

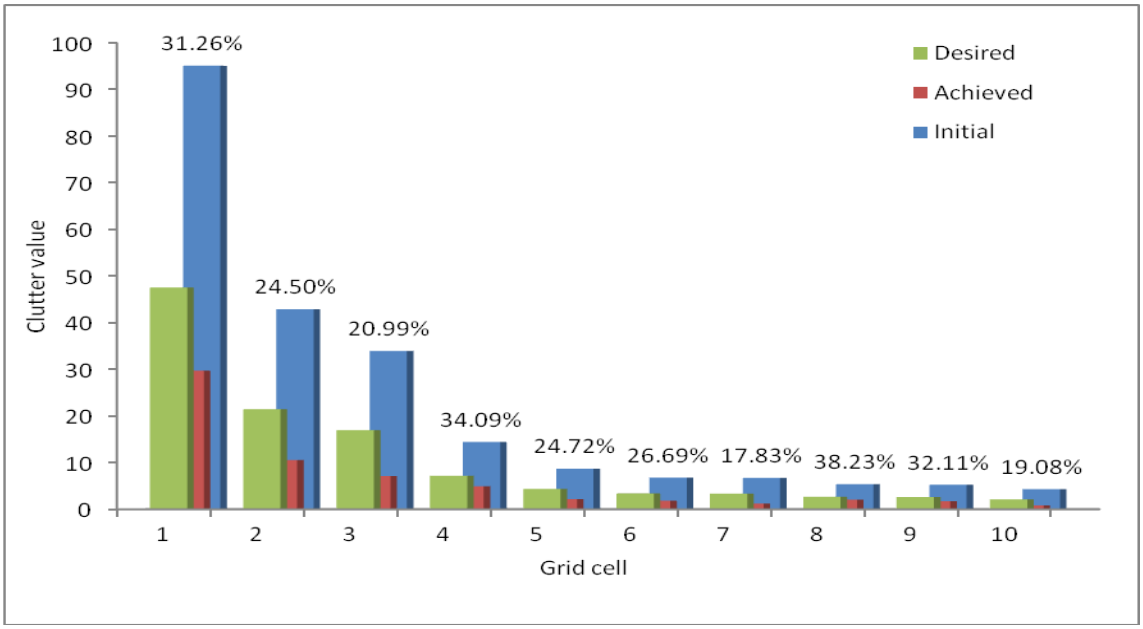


Figure 25 – clutter values obtained after dataset reduction CC and line simplification

5.3. Medium dataset

Increasing the size of the dataset means that operating on an individual trajectory (removal or simplification) has less impact on the global function; this allows the dataset reduction methods to remove more trajectories before they reach the maximum value. The three methods are illustrated in the figure below.

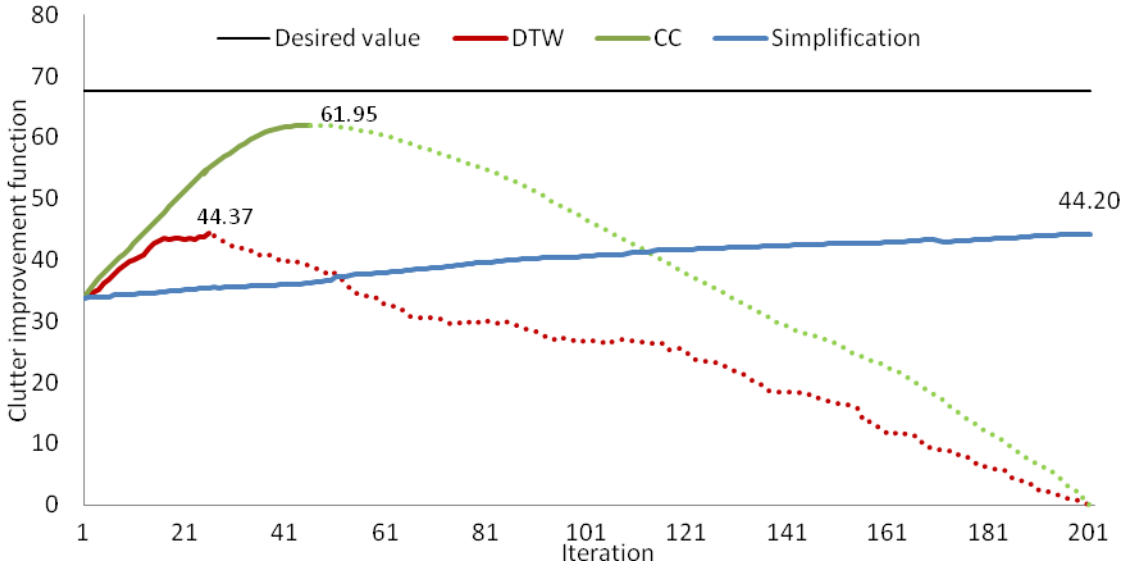


Figure 26 - The three methods compared, applied on a subset of 200 trajectories.

Aside from the fact that the data reduction methods remove almost 50% more trajectories from the 200-set than from the small set, they perform in the same way. Removal using the clutter contribution measure brings the global function very close to the desired value and it lowers clutter proportionally, while using the dynamic time warping distance results in concentrated removal from the most cluttered grid cells.

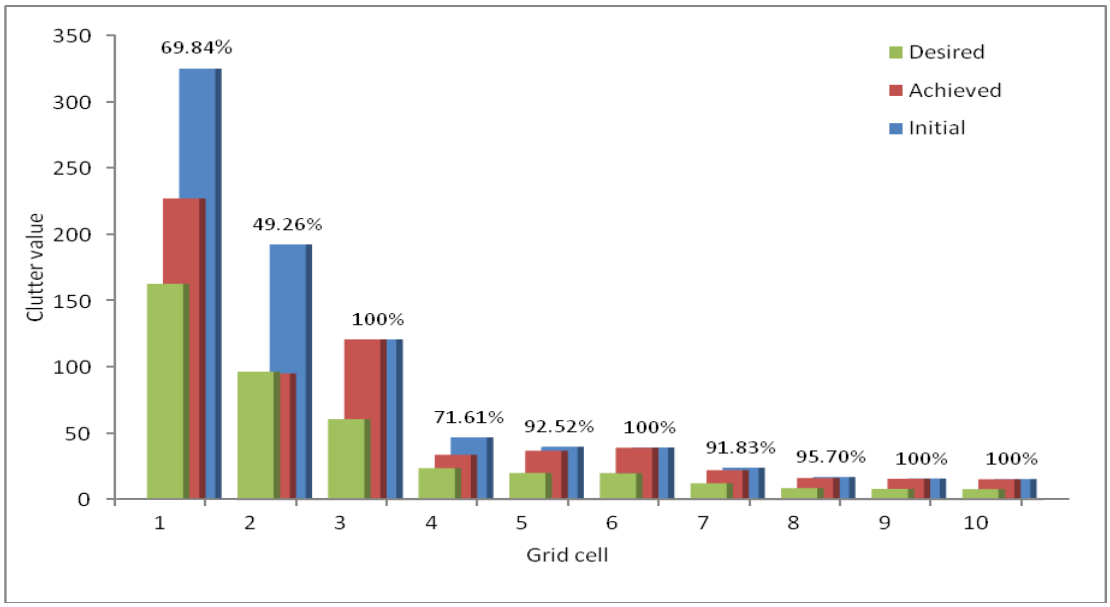


Figure 27 – clutter values when applying dataset reduction using DTW on a 200-dataset

The two methods that have displayed the best performance when applied to the small dataset give the best global function value for the medium dataset as well.

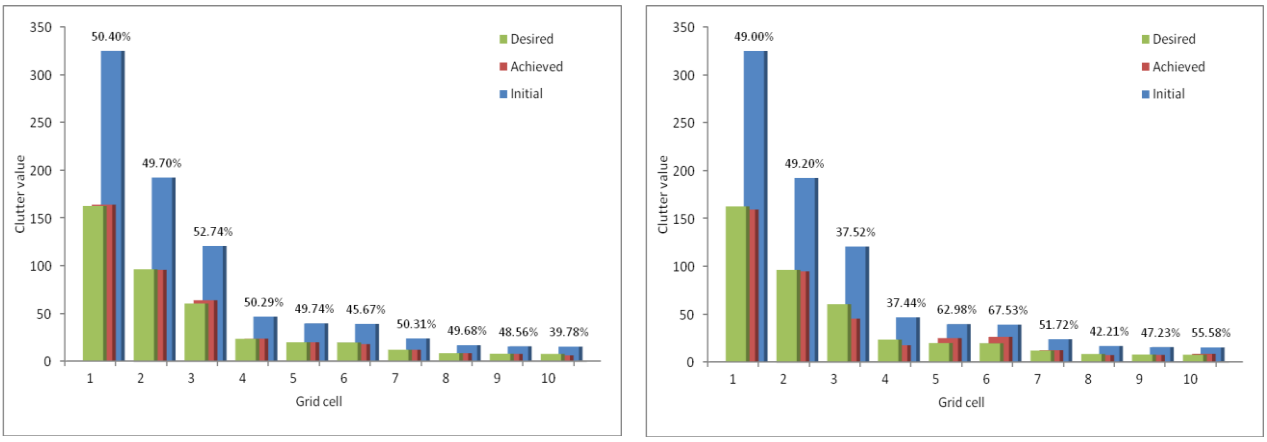


Figure 28 – Clutter values for reduction using clutter contribution, on the left, and simplification followed by reduction CC, on the right.

The methods have also been applied to the dataset of 500 trajectories shown in Figure 8. The methods prove to be consistent and the graphs plotting the clutter improvement function are very similar. They are presented in the Appendix.

5.4. Comparison with random removal

The dataset reduction method operates according to set rules: at each iteration it uses one of the two criteria for determining which trajectory can be removed from the set. But how does that compare to a random selection of trajectories to be removed?

For the first dataset of 200 trajectories, dataset reduction using dynamic time warping started with a global value of 41.64 and recorded a maximum clutter improvement function of 52.96 from removing 39 trajectories. Random removal of 39 trajectories from the dataset resulted in an average global function of 62.35, which is considerably higher towards the desired 83.29. However, the clutter reduction is random, some grid cells getting less clutter than desired, some virtually none at all, as illustrated in figure 29.

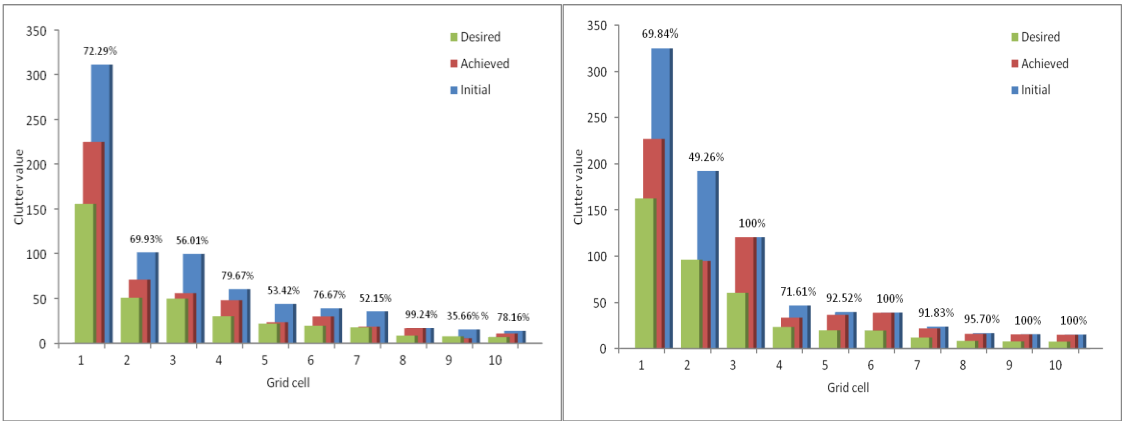


Figure 29 – Local clutter reduction for random removal (left) and dynamic time warping (right)

Reduction using the clutter contribution measure performs better in every sense than doing dataset reduction with random selection. The algorithm manages to remove 45 trajectories from the dataset of 200, achieving a global clutter value of 79.45, close to the desired 83.29. On average, random removal of the same number of trajectories offers an average global of 67.11 and displays the same randomness on a local level (figure 30).

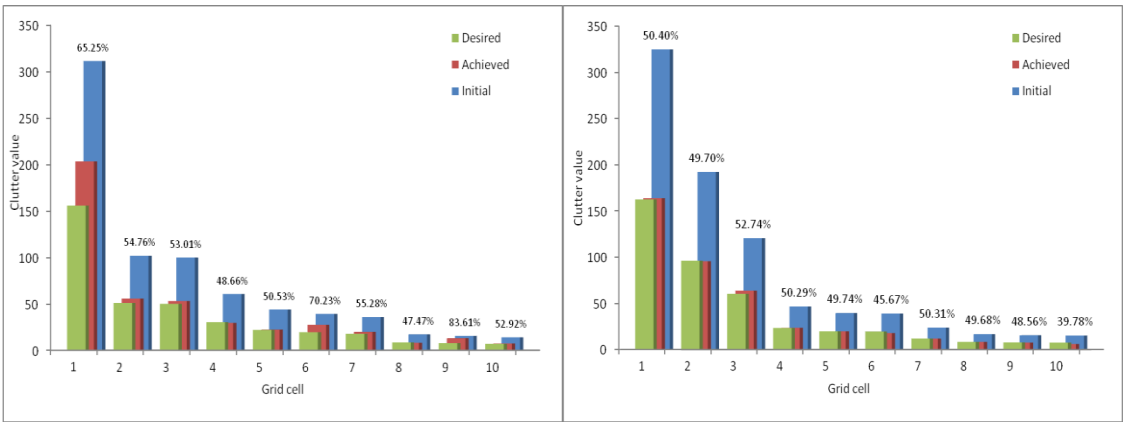


Figure 30 - Local clutter reduction for a random removal (left) and clutter contribution (right)

Figure 31 – Dataset of 500 trajectories reduced: above, with random choice and below, using the clutter contribution measure

5.5. Parameter tweaking

The main experiments have been run using a set of parameters whose values have been empirically determined. The grid superimposed on the visualization is made up of 10x10 square cells. The length of an edge of a grid cell is used for the values of constants *c* and *c'* in the local clutter formula $C = L + c \cdot N + c' \cdot I$, for scaling. Furthermore, the global clutter improvement function uses a fraction parameter to determine the desired clutter in each cell compared to the initial values. The main experiments aim for 50% of the initial clutter. Lastly, the contextual line simplification algorithm calculates allowed shortcuts in a trajectory based on a maximum distance ε, which has been set to 1/200 of the edge of a grid cell. Varying these parameters gives an indication of how the algorithms can be adapted for better performance or for datasets with special features.

Error tolerance ε

The contextual line simplification algorithm is applied on all trajectories in a dataset and is dependent on only one parameter, which is the maximum allowed distance at which a shortcut can pass from the omitted points. Varying this tolerance gives a greater reduction of intersection points throughout the dataset and subsequently a greater improvement of the global clutter function. Table 1 gives an overview of the varying performance of the line simplification algorithm for ε being set to various fractions of the grid edge; the ideal achieved clutter is 50%.

Epsilon value	**1/500**	**1/400**	**1/200**	**1/100**	**1/50**
Intersections removed	7.55%	9.70%	20.70%	33.96%	48.86%
Clutter achieved	96.38%	95.39%	89.04%	78.68%	68.86%

Table 1 - varying the error tolerance ε

Clutter reduction factor

The clutter reduction factor represents the percentage of clutter that is desired in each grid cell, compared to the original. For the main experiments, this factor has been set to 50%. Other reduction factors that were tested include 75%, 25% and 10%. The line simplification algorithm is independent of this factor, since it only depends on the error ε above, but the dataset reduction algorithms operate with the global clutter improvement function. A greater reduction factor means the desired value for the global function is proportionally larger than the initial one, forcing the algorithm to remove a greater number of trajectories to meet the target. The tests varying the clutter reduction factors were performed on a dataset of 200 trajectories and are focused on the global improvement, not local. The three main methods have predictable evolutions when setting a smaller desired value for the clutter.

Contextual line simplification uses the same distance measure across all these tests, so its performance becomes increasingly worse in relation to the desired global clutter. For reducing clutter to 25%, simplification achieves no more than 90%, which means that for removing more clutter the ε distance has to be increased.

Dataset reduction using dynamic time warping also sees its performance decay over the four sets of tests, proving that the current method is not scalable in its current form. The clutter is reduced to 91.5% for the first case but only 73% for the last test, as shown in the table below.

The second reduction measure using clutter contribution registers only slight performance decay, with the achieved clutter reduction always within 10% of the desired factor. The downside is that for achieving a 90% reduction of clutter it is necessary to remove more than half of the set.

All results from setting different clutter level goals are presented in table 2.

Method / %clutter desired	Reduction with DTW		Reduction with CCM		Line simplification
	Achieved	Removed	Achieved	Removed	Achieved
75%	91.49%	6	79.23%	20	84.43%
50%	84.38%	25	58.38%	45	84.63%
25%	77.47%	59	34.02%	80	91.58%
10%	73%	84	26.63%	118	96.63%

Table 2 - varying the clutter reduction percentage

5.6. Overview of method performance

Reducing clutter by half in each grid cell has been investigated by developing two dataset reduction techniques and a line simplification algorithm. Table 3 contains the performance numbers averaged across all datasets tested, including the combination of line simplification and dataset reduction. The last column represents the percentage by which global clutter is lowered, with an ideal value being 50%.

Method	% trajectories removed	% intersections removed	% global clutter improvement
Reduction DTW	15.24%	36.11%	83.99%
Reduction CC	21.52%	51.36%	57.12%
Simplification	-	36.20%	76.59%
Simplification and Reduction DTW	4.68%	48.42%	70.65%
Simplification and Reduction CC	10.48%	52.73%	58.92%

Table 3 – global percentages for the five methods or combination of methods

The results vary with the number of trajectories and it is also influenced by the type of dataset used. For the first case, all methods that involve dataset reduction using the clutter contribution measure perform *better* with increasing datasets, even though it removes a *smaller* percentage of the trajectories. Specifically, when operating on the first subsets of 100, 200 and 500 trajectories, CCM (clutter contribution measure) achieves 58.5%, 54.6% and 53% clutter improvement.

6. Conclusions

The experiments carried out in the previous chapter have been consistent across all datasets of different sizes. They bring forth some clear results.

Firstly, when attempting to optimize the clutter improvement function defined in subchapter 3.1, the method that performed best is dataset reduction using the clutter contribution measure, which removed an average of 22% of trajectories to reach an average of 57.12% of the initial clutter levels. A very close second is the combination between line simplification and dataset reduction using the same clutter contribution measure, which performs 2% worse but removes only 10% of the trajectories. Consequently, if data integrity represents a priority, the above combination represents the best choice for improving clutter values.

Incremental dataset reduction using the dynamic time warping distance has been an experimental measure to assess how human perception about clutter performs, given the set measure for clutter. Intuitively, the trajectory "closest" to a whole dataset represents a cluster prototype, so removing it would increase the distances between remaining trajectories. It is also more likely that a trajectory that's "close" to another trajectory or group of trajectories will have more intersections with them. In practice, this thesis has shown that dynamic time warping performs best at lowering clutter *locally* within the most cluttered sections of a dataset, while the clutter contribution measure optimizes clutter *globally*. Another advantage of the dynamic time warping measure is that it removes trajectories from where they are more concentrated; however, this is based on the assumption that most datasets have trajectories concentrated towards the center of the map. The obvious disadvantage is that the DTW measure doesn't minimize intersection points and has shown to be ineffective at a global scale.

Performing dataset reduction by removing the trajectories that contribute most to clutter (through length and intersection points) is the most effective at maximizing the local as well as the global measures of clutter introduced in this thesis. The method is also scalable to larger datasets and does not oversimplify the set.

Line simplification in the context of intersection points has shown to offer a constant reduction in clutter through minimizing the number of intersection points between trajectories. For a small enough value of the ε distance, trajectories suffer minor modifications in length. If the clutter reduction factor is set to a higher value, the algorithm can be adapted by increasing ε.

However, line simplification by itself will not lower clutter to a set proportion; a combination that has been experimentally shown to perform well is an initial run of line simplification, minimizing the intersection points, followed by dataset reduction using the best method, clutter contribution. The result has several unique characteristics: intersection points removed proportionally across the entire visualization, fewer trajectories removed from the dataset, as well as lowering the clutter values both locally and globally.

Even though this thesis has introduced an objective measure for clutter in the visualization of a dataset, the idea of improving a visualization is subjective. Depending on user preferences or application purpose, it might be more suitable to lower clutter locally with a slightly modified dynamic time warping measure. If data integrity is a priority, then line simplification followed by dataset reduction using the clutter contribution measure is the best option. Otherwise, for the best improvement of clutter globally across the visualization, the dataset reduction measure presented above offers the best results.

The measure of clutter, as well as the two algorithms for the improvement of the visualization, approach a novel direction in the research area of trajectory visualization optimization, building

on the work of Janssen and van Kreveld [35]. Compared to Rosenholtz *et al.* [36], which uses contrast and luminance, I focus on perennial attributes of the trajectories, such as geometrical shape and sample points. The clutter measure is also designed to be absolute, without alteration at different levels of detail. Experiments have shown that the methods have specific advantages and bring significant improvement to the visualization.

7. Future work

The goal of this thesis was to research how a number of data-processing methods can improve the visualization of a dataset of spatial trajectories. There are plenty of limitations to the approach. Only data-altering methods are being used, without resorting to styling or projections in different coordinate systems. Unlike Janssen and van Kreveld [35], color and contrast are not considered as clutter-inducing factors, which can be debatable. I consider that, given a dataset of trajectories, post-processing can indeed improve "readability" and understanding of a map, but it doesn't constitute a permanent improvement to the dataset and it can vary from user to user.

Also, several follow-up improvements can be made to the research presented in this thesis. A formal user study can be of great help in determining the correlation between the clutter improvements measured by my methods and how human users perceive different levels of clutter.

An important part of the algorithm that hasn't been investigated is how to set a desired level of clutter for a bounding box optimally. This would translate to the very difficult task of determining an optimum amount of clutter relative to a specified input and possibly additional user preferences. In the same category of possible improvements, testing a trajectory for self-intersections can also be of use in reducing effective clutter. Another future approach is to alter the incremental dataset reduction using the dynamic time warping distance so that it performs better for the set clutter measure.

Contextual line simplification is an algorithm that can be applied multiple times, but due to the lack of time the experiments haven't been performed. Alternatively, chapter 5.5 has shown that varying different parameters help to further reduce the clutter locally as well as globally. An investigation into how the different parameters can be balanced for best improvement would be useful.

Finally, the present implementation of the algorithms can be further improved so it could be applied practically to even larger datasets.

8. Bibliography

Surveys of visualization techniques

1. Group discussion with K. Buchin, U. Demšar, A. Slingsby and E. Willems. *Results of the break-out group: Visualization*, Dagstuhl Seminar Proceedings, http://drops.dagstuhl.de/opus/volltexte/2011/2986

2. G. Andrienko, N. Andrienko, *Visual Analytics of Movement*. ACM SIGKDD Explorations Newsletter - Special issue on visual analytics, Volume 9 Issue 2, December 2007

3. N. H. Long, W*eb Visualization of Trajectory Data using Web Open Source Visualization Libraries*, Master thesis, 2010.

Trajectory data analysis and mining

4. A. Monreale, G. Andrienko, N. Andrienko, F. Giannotti, D. Pedreschi, S. Rinzivillo, S. Wrobel. *Movement Data Anonymity through Generalization*. Transactions on data privacy 3 (2010), pages 91–121.

5. J. Lee, J. Han, X. Li, *Trajectory Outlier Detection: A Partition-and-Detect Framework*. ICDE '08 Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, Pages 140-149.

6. G. Andrienko, N. Andrienko, S. Wrobel, *Visual Analytics Tools for Analysis of Movement Data*. ACM SIGKDD Explorations Newsletter - Special issue on visual analytics, Volume 9 Issue 2, December 2007, Pages 38-46

7. T. Schreck, J. Bernard, T. Tekušová, J. Kohlhammer. V*isual Cluster Analysis of Trajectory Data With Interactive Kohonen Maps*. Journal on Information Visualization, Volume 8 Issue 1, January 2009, Pages 14-29

8. N Pelekis, G Andrienko, N. Andrienko, Ioannis Kopanakis, Gerasimos Marketos, Yannis Theodoridis. Visually exploring movement data via similarity-based analysis, Journal of Intelligent Information Systems, April 2012, Volume 38, Issue 2, pp 343-391

Graph visualization

9. F. Van Ham and M. Wattenberg. *Centrality Based Visualization of Small World Graphs*, EUROGRAPHICS 2008, Volume 27 (2008), Number 3

10. D. Holten and J. Van Wijk. *Force-Directed Edge Bundling for Graph Visualization*. Eurographics/ IEEE-VGTC Symposium on Visualization 2009, Volume 28 (2009), Number 3

11. A. Telea and O. Ersoy. *Image-Based Edge Bundles: Simplified Visualization of Large Graphs*, Eurographics/ IEEE-VGTC Symposium on Visualization 2010, Volume 29 (2010), Number 3

12. V. Batagelj, W. Didimo, G. Liotta, P. Palladino, M. Patrignani. *Visual Analysis of Large Graphs Using (X;Y)-clustering and Hybrid Visualization*. In Proc. IEEE Pacific Visualization 2010, Page(s): 209 – 216.

Density maps

13. R. Scheepens, N. Willems, H. Van De Wetering, G. Andrienko, N. Andrienko, J. J. Van Wijk. *Composite Density Maps for Multivariate Trajectories.* Visualization and Computer Graphics, IEEE Transactions on Visualization and Computer Graphics, Volume: 17 Issue: 12, pages 2518 – 2527

14. R. Scheepens. G*PU-Based track visualization of multivariate moving object data*. Master thesis 2010

15. R. Scheepens N. Willems, H. Van De Wetering, J. Van Wijk. *Interactive Visualization of Multivariate Trajectory Data with Density Maps*. IEEE Pacific Visualization Symposium 2011, Page(s): 147 - 154

16. M. D'auria, M. Nanni, and D. Pedreschi. *Time-focused density-based clustering of trajectories of moving objects*. Journal of Intelligent Information Systems, Volume 27, Number 3, 267-289.

17. N. Willems. *Visualization of Vessel Traffic*. Proefschrift, 2011.

Volumetric approaches

18. G. Andrienko, N. Andrienko. *A Visual Analytics Approach to Exploration of Large Amounts of Movement Data*. Lecture Notes in Computer Science, 2008, Volume 5188/2008, 1-4.

19. M. Schirski, T. Kuhlen, M. Hopp, P. Adomeit, S. Pischinger, C. Bischof. *Efficient Visualization of Large Amounts of Particle Trajectories in Virtual Environments Using Virtual Tubelets*. Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry Pages 141 – 147.

20. G. Albuquerque, M. Eisemann, D. Lehmann, H. Theisel, M. Magnor. Im*proving the Visual Analysis of High-dimensional Datasets Using Quality Measures*. Proc. IEEE Symposium on Visual Analytics Science and Technology (VAST), Salt Lake City, Utah, USA, pp. 19–26.

21. Z. Wang, H. Guo, B. Yu, X. Yuan. I*nteractive Visualization of 160 Years' Global Hurricane Trajectory Data*. Proceedings of IEEE Pacific Visualization Symposium (PacificVis 2011), pages 37-38, Hong Kong, March 1-4, 2011.

22. G. Andrienko, N. Andrienko, S. Rinzivillo, M. Nanni, D. Pedreschi, F. Giannotti. I*nteractive Visual Clustering of Large Collections of Trajectories*. Visual Analytics Science and Technology, 2009. VAST 2009. Issue Date: 12-13 Oct. 2009, on page(s): 3 – 10

Visualization systems

24. Z. Yan, L. Spremic, D. Chakraborty, C. Parent, S. Spaccapietra, K. Aberer. *Automatic Construction and Multi-level Visualization of Semantic Trajectories*. Proceedings of

the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems Pages 524-525.

25. C. Hurter, B. Tissoires, S. Conversy. *FromDaDy: Spreading Aircraft Trajectories Across Views to Support Iterative Queries*. Visualization and Computer Graphics, IEEE Transactions on Visualization and Computer Graphics, Issue Date: Nov.-Dec. 2009, Volume: 15 Issue:6, on page(s): 1017 – 1024

26. C. Hurter, O. Ersoy, A. Telea, *MoleView: An Attribute and Structure-Based Semantic Lens for Large Element-Based Plots*. IEEE transactions on Visualization and Computer Graphics, December 2011 (vol. 17 no. 12) pp. 2600-2609

27. X. Wang, K.T. Ma, G. Ng. *Trajectory Analysis and Semantic Region Modeling Using Nonparametric Hierarchical Bayesian Models*. International Journal of Computer Vision Volume 95, Number 3, 287-312.

28. H. Guo, Z. Wang, B. Yu, H. Zhao, X. Yuan. *TripVista: Triple Perspective Visual Trajectory Analytics and Its Application on Microscopic Traffic Data at a Road Intersection*. In Proceedings of IEEE Pacific Visualization Symposium (PacificVis 2011), pages 163-170, Hong Kong, March 1-4, 2011.

29. O.D. Lampe, J. Kehrer, H. Hauser. *Visual Analysis of Multivariate Movement Data using Interactive Difference Views*. Proceedings of Vision, Modeling, and Visualization (VMV 2010), pages: 315–322

Domain-specific visualizations

30. N. Willems, H. Van De Wetering, J. J. Van Wijk. *Evaluation of the Visibility of Vessel Movement Features in Trajectory Visualizations*. Eurographics / IEEE Symposium on Visualization 2011 (EuroVis 2011), Volume 30 (2011), Number 3

31. E. Grundy, M. Jones, R. Laramee, R. Wilson, E. Shepard. *Visualisation of Sensor Data from Animal Movement*. Eurographics/ IEEE-VGTC Symposium on Visualization 2009, Volume 28 (2009), Number 3

General

32. N. Andrienko, G. Andrienko, N. Pelekis, S. Spaccapietra. *Chapter 2 - Basic Concepts of Movement Data*. Mobility Data Mining and Privacy Geographic Knowledge Discovery (2008), Pages: 15-38

Clutter

33. R.J. Phillips. *An Investigation of Visual Clutter in the Topographic Base of a Geological Map*. The Cartographic Journal, Volume 19, No. 2, December 1982

34. W. Peng, M. O. Ward, E. A. Rundensteiner. *Clutter Reduction in Multi-Dimensional Data Visualization Using Dimension Reordering*. IEEE Symposium on Information Visualization 2004

35. M. Jansen, M. van Kreveld. *Evaluating the Consistency of Cartographic Generalization*. In T.K. Poiker & N. Chrisman (Eds.), Proceedings 8th International Symposium on Spatial Data Handling (pp. 668-678). Burnaby, Canada: GIS LAB, Simon Fraser University.

36. R. Rosenholtz, Y. Li, J. Mansfield, Z. Jin. *Feature Congestion: A Measure of Display Clutter*. CHI '05 Proceedings of the SIGCHI conference on Human factors in computing systems, pages 761 - 770

37. R. Rosenholtz, Y. Li, L. Nakano. *Measuring visual clutter*. Journal of Vision (2007) 7(2):17, 1–22

38. M. Jansen. *The Clutter Function, an evaluation tool for automated map generalization*. INF-SCR-98-01. Februari 1998

Dynamic Time Warping

39. K. Wang, T. Gasser. *Alignment Of Curves By Dynamic Time Warping*. The Annals of Statistics 1997, Vol. 25, No. 3, 1251-1276

40. R. Niels. *Dynamic Time Warping: An intuitive way of handwriting recognition?* MASTER THESIS, November/December 2004, Radboud University Nijmegen

41. P. Senin. *Dynamic Time Warping Algorithm Review*. University of Hawaii at Manoa, Honolulu, USA, December 2008

Sweep Line Algorithm

42. M. Smid. *Computing intersections in a set of line segments: the Bentley-Ottmann algorithm*, lecture notes at the School of Computer Science, Carleton University, Ottawa, Ontario, 2003.

Trajectory Data

43. GeoLife GPS Trajectories, *http://research.microsoft.com/en-us/downloads/b16d359d-d164-469e-9fd4-daa38f2b2e13* , Microsoft Research.

44. Yu Zheng, Lizhu Zhang, Xing Xie, Wei-Ying Ma. *Mining interesting locations and travel sequences from GPS trajectories*. In Proceedings of International conference on World Wild Web (WWW 2009), Madrid Spain. ACM Press: 791-800.

45. Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, Wei-Ying Ma. *Understanding Mobility Based on GPS Data*. In Proceedings of ACM conference on Ubiquitous Computing (UbiComp 2008), Seoul, Korea. ACM Press: 312-321.

47. Mark de Berg, Otfried Cheong, Marc van Kreveld, Mark Overmars, *Computational Geometry: Algorithms and Applications*, Springer-Verlag, 3rd rev. ed. 2008.

Line Simplification

48. H. Imai and M. Iri, *Computational-Geometric Methods for Polygonal Approximation of a Curve*, Computer Vision, Graphics and Image Processing, 36, pp. 31--41, 1986.

49. Hiroshi Imai and Masao Iri, *Polygonal Approximations of a Curve – Formulations and Algorithms*, Computational Morphology p. 71-86, 1988

50. W.S. Chan and F. Chin, *Approximation of Polygonal Curves with Minimum Number of Line Segments*, International Journal of Computational Geometry and Applications, 1996, v. 6 n. 1, p. 59-77