Identifying Locations from Geospatial Trajectories

Alasdair Thomason[a,*], Nathan Griffiths[a], Victor Sanchez[a]

[a] *Department of Computer Science, University of Warwick, Coventry, CV4 7AL, United Kingdom*

Abstract

Harnessing the latent knowledge present in geospatial trajectories allows for the potential to revolutionise our understanding of behaviour. This paper discusses one component of such analysis, namely the extraction of significant locations. Specifically, we: (i) present the Gradient-based Visit Extractor (GVE) algorithm capable of extracting periods of low mobility from geospatial data, while maintaining resilience to noise, and addressing the drawbacks of existing techniques, (ii) provide a comprehensive analysis of the properties of these visits and consequent locations, extracted through clustering, and (iii) demonstrate the applicability of GVE to the problem of visit extraction with respect to representative use-cases.

Keywords: Geospatial, Location Extraction, Significant Locations, Trajectory, Visits

1. Introduction

The recent rise in the prevalence of location-aware hardware has brought with it an increase in the availability of geospatial trajectories, along with the consequent foundation for reasoning about the actions of users that this affords. Fundamentally, such trajectories are sets of spatio-temporal datapoints that relate the whereabouts of an individual, animal or other entity to specific times. The extraction of meaningful locations from geospatial data provides a basis for modelling a user's interactions with their environment, an important part of many location-aware applications. Such applications may include location prediction, typically relying on extracted locations to discretise the set of possible predictive outcomes, or context-aware services such as recommender systems or digital assistants that use location to provide a greater level of personalisation.

This paper considers location extraction as a two-step process that first extracts periods of low mobility from geospatial data, referred to as *visits*, and then clusters these visits together to form *locations*. Specifically, the paper provides a detailed discussion of existing techniques for *visit extraction* and *visit clustering*, and presents the Gradient-based Visit Extraction (GVE) algorithm for the purpose of visit extraction. This algorithm provides resilience to noise and overcomes several drawbacks of previous works: the algorithm does not place a minimum bound on visit duration, has no assumption of evenly spaced observations and operates in real-time without imposing a delay on trajectory points being considered. Additionally, we provide a thorough exploration of the parameter space for GVE through a comprehensive analysis of the properties of visits and locations produced, and demonstrate the applicability of GVE to the task of visit extraction with respect to representative use-cases and a comparison to the current state-of-the-art.

The remainder of this section discusses visit extraction and proposes two representative use-cases for extracted visits and locations. Section 2 then goes on to discuss related work, including approaches to visit

*Corresponding author

Email addresses: ali@dcs.warwick.ac.uk (Alasdair Thomason), nathan@dcs.warwick.ac.uk (Nathan Griffiths), vsanchez@dcs.warwick.ac.uk (Victor Sanchez)

extraction and clustering. The GVE algorithm is presented and discussed in Section 3, and discussion of the methodology employed to analyse the algorithm and the visits it produces can be found in Section 4. Finally, Section 5 details the results of the analysis and demonstrates the applicability of GVE to the task of visit extraction, with Section 6 providing a conclusion and summary of the findings and contributions made.

1.1. Visit Extraction

The extraction of locations meaningful to users is achieved by analysing the datapoints found within trajectories and identifying the regions where the user has spent time. Although a variety of techniques have been used in literature to extract locations from data, they are typically used as a precursor step to performing another activity, such as location prediction, and have not been investigated or evaluated in depth. While it is possible to perform location extraction by using a single clustering algorithm, the process is typically performed in two distinct clustering steps [1–3, 7, 24, 28]. The first step, *visit extraction*, is responsible for partitioning a temporally ordered dataset into periods of low mobility, referred to as either *stops*, *stays* or *visits*, where during each period the user is expected to have remained in one geographic location. For clarity, in this paper, we refer to such periods as *visits*. Visits of no duration (i.e. either the visit consists of a single point, or all points that make up a visit were recorded at the same time instance) are classed as noise since they represent the user travelling and not being stationary. The second step in the process, *visit clustering*, summarises the extracted visits and performs clustering to determine which visits belong to the same *location*.

Utilising a two-step approach for location extraction has several advantages, namely:

- Visit extraction can be performed in linear time, summarising vast portions of the dataset, thus reducing the complexity for the clustering step.
- By considering their temporal nature, individual data points that occur when an entity is moving are ignored. In traditional clustering, if several points were to be recorded in close proximity, but on different occasions, for instance along a road, an erroneous location would be identified.
- Extracted visits consist of contiguous points and thus characterise a period of time in which the user remained at the location, providing a basis for modelling historic time spent.

The disadvantages of the two-step process relate to the location clusters extracted at the visit clustering step. In order to reduce the complexity of this stage, extracted visits are typically summarised into a single point (e.g. centroid), and consequently the shape of overall locations extracted are not likely to be represented. Depending upon the goal of location clustering, this could be problematic.

1.2. Use-cases

The uses for extracted locations and visits are varied as they provide a foundation for modelling behaviour. However, for the remainder of this paper we consider two representative examples, which apply equally well to trajectories from various sources. The first use-case, referred to as *accurate visits*, considers the visits as a source of context, aiming to accurately characterise a user's physical location at any point in time. In this case, clustered locations primarily serve to group the visits together to model transitions correctly. The second use-case, *location properties*, is less focused on visits, but rather, considers the accurate identification of the properties (i.e. shape and position) of locations. Accurately identified locations are essential to certain services, such as creating geofences, where the visits are of less importance. It is important to note, however, that although the *location properties* use-case does not strictly require the accurate extraction of visits, the runtime of visit clustering algorithms is severely detrimented as the number of visits increases.

2. Related Work

Treating location extraction from trajectories as a two-step process is a technique that has been employed in literature [1–3, 7, 24, 28]. One of the earliest examples is an investigation conducted by Ashbrook and Starner into identifying significant locations, from a dataset of GPS points, with the aim of predicting user movement [3, 4]. From the collected data, Ashbrook and Starner observed that the data loggers used did not function well indoors, as a GPS signal was rarely available, and therefore treated periods of missing data as visits. Once extracted, these visits were clustered together using the k-means clustering algorithm, selecting an appropriate value for k by performing the clustering multiple times on different parameters and observing the results. The particular algorithms employed here have their own respective drawbacks, and it is the focus of the remainder of this section to explore these drawbacks along with alternative approaches and related work. Towards understanding existing approaches to visit extraction and clustering, Section 2.1 begins with a discussion on the collection of data over which visit extraction can be performed. An overview of visit extraction techniques follows in Section 2.2, and visit clustering in Section 2.3. Finally, Section 2.4 explores work that has been conducted into enriching extracted locations with additional meaning and context information.

2.1. Data Collection

Before discussing methods of location extraction, it is important to consider the collection and processing of data, to gain an understanding of the properties expected when the location extraction stage is reached. In much existing research into location-aware systems, users are asked to carry battery-operated devices which record their location at predetermined intervals [3, 19, 24, 28, 31]. This approach places a burden upon the users who must ensure that the device remains with them and is charged at all times. In part, these drawbacks have been reduced in recent years due to the inclusion of location-sensing hardware in personal smartphones, and thus recent location-aware literature has used data from such devices [7, 9, 18, 22, 24, 29]. Location is determined by the device through a combination of sensors, such as GPS, GLONASS and cell tower triangulation, where the accuracy of measurement is strongly correlated with the amount of power consumed.

Aiming to reduce the power overhead of data collection, Kiukkonen et al. present a state-based machine that transitions between different rates and location determination methods based on sensor readings [19]. Specifically, the authors use WiFi base stations to indicate locations when available, and adjust collection rate based on accelerometer readings (i.e. if the user is moving, the collection rate is increased). Similarly, Chon et al. present an application, SmartDC, that aims to estimate when a user will leave a current area and increase collection around this time [10]. As a consequence of techniques such as this, and the limitations of these devices, it is important to note that the accuracy and rate of location collection will not remain constant.

2.2. Visit Extraction Techniques

Ashbrook and Starner's approach, namely that of looking for periods of missing data within trajectories, is limited in that it assumes that all missing data is caused by a visit, and visits cannot occur when data was collected. Indeed, the authors note that the data logging devices were prone to run out of battery power, also causing a lack of data. Building on this work, but assuming a constant flow of data, even when indoors, similar algorithms have been proposed. Relying on time and distance thresholds, the algorithms typically operate by iterating over a temporally ordered dataset and segmenting the data into visits, where no visit can be larger than a specified radius (or, sometimes, that no consecutive points can be greater than a specified distance apart) and the user must remain in this area for longer than a specified amount of time [1, 2, 17, 18, 28, 31, 33]. This approach was initially presented by Li et al., but has since been adapted for specific uses in literature [21]. Montoliu and Gatica-Perez also use this technique, but add an additional constraint that the time between consecutive data points in the same visit must be bounded [24]. This was proposed to prevent periods of missing data from being contained within a visit. If data became unavailable at one time, and became available at a nearby coordinate some time later, it is not possible to state with certainty that the user remained stationary for the missing period.

Although using maximum radius and minimum duration thresholds may overcome the issues caused by assuming that a loss of GPS signal is equivalent to a visit, they all suffer from a lack of resilience to noise and, with the exception of Montoliu and Gatica-Perez, do not consider missing data. In all cases, the algorithms consider a single point and ask whether this point belongs to the previous visit, or signals the end of the previous visit and the start of a new one. The problem with this approach is that all location detection techniques are prone to noise to varying degrees, and while a single point may occur outside the preselected distance threshold (e.g. the distance from the centre of the already extracted visit), this could be due to noise in the dataset. Furthermore, while the results of some of the techniques used have been demonstrated over a single small example, no quantitative evaluation of the properties of the visits extracted has been conducted.

In contrast to the approach of splitting a temporally ordered dataset into visits, there are methods of extracting visits that are not partitional in nature. ST-DBSCAN [8] and DJ-Cluster [32] are density-based clustering algorithms that are designed to cater for spatio-temporal data by extracting clusters that are similar in both space and time. While they overcome the problem of performing only two-dimensional clustering directly to trajectories (that of extracting dense groups of points without considering time), these approaches are computationally intensive, typically having runtime $O(n \log n)$ as opposed to the $O(n)$ of the other techniques discussed, and the resultant visits do not span continuous time periods.

Aiming to overcome the drawbacks of existing approaches, by assuming noise in the dataset, Bamis and Savvides present the STA extraction algorithm [6]. While the authors were specifically motivated by identifying activities that repeat in cycles through extraction and clustering, the first step, STA extraction, uses a definition of an activity that is identical to our definition of a visit, and thus performs visit extraction. The algorithm is similar to existing approaches in that it iterates over the trajectory points, but uses a weighted averaging filter to reduce the impact of noise before considering an activity as having ended. This technique however does bring with it several drawbacks and assumptions relating to the data, for example, requiring evenly time-sliced data and a full buffer before consideration of a visit can occur, consequently imposing a minimum bound on visit duration.

2.3. Visit Clustering

Extracted visits can be clustered together to form locations. Several general-purpose clustering techniques exist and have been used for this purpose, including k-means [3, 23] and DBSCAN [1, 5, 12, 24]. When clustering with k-means, the number of clusters to be extracted must be known a priori, which is not typical when extracting locations from trajectories. DBSCAN, on the other hand, has no such requirement, instead extracting clusters providing that a sufficient density of data points exist.

Focusing specifically on location extraction, Zheng et al. [28] propose a grid-based clustering algorithm that splits the considered area into uniform squares, where the size of each square is a user-specified parameter. Squares that contain a greater number of visits than a threshold are then merged with all of their neighbours, providing that no dimension exceeds 3 squares in length. This requirement, although bounding location size, enforces the requirement for the maximum size of locations to be known a priori, and creates locations that are regular in shape. These locations may, therefore, not represent the shape of real-world locations accurately, and thus the algorithm is likely not applicable when considering the *location properties* use-case presented previously.

Finally, continuing on from their STA extraction algorithm, discussed in Section 2.2, Bamis and Savvides propose the STA clustering algorithm for the detection of activities that were performed in similar locations and at similar times of day. The STA clustering algorithm operates by summarising visits as their 3-dimensional bounding box (along the latitude, longitude and time dimensions) and progressively merging visits into clusters based on a similarity function, with weightings given to longitude, latitude and time. While it is possible to ignore the time dimension, as we would wish for location extraction, by setting the weighting to 0 the algorithm is not designed to merge all overlapping visits, only those whose dimensions are above a certain similarity, resulting in locations that can overlap. While the STA extraction algorithm is applicable for both extracting visits and activities, the STA clustering algorithm is not ideally suited to the task of visit clustering.

2.4. Enriching Locations

Once extracted, locations can be used as a basis for many location-aware systems and services, for example, to guide individuals around a city or other point of interest [27, 28, 30]. Further applications for extracted locations include categorising locations through supervised learning [1, 2, 11, 13], judging user similarity by location transitions [16], and the identification of activities from visited locations [5].

In addition to these uses, location prediction has been a key focus of research, aiming to identify which locations a user will likely visit in the future. Ashbrook and Starner present one of the earlier pieces of work in this area, using Markov models to model transitions between extracted locations [3, 4]. Building upon this, and considering time as well as the current location, Wang and Prabhala propose a combined transition and periodicity-based approach [26], and Fukano et al. investigate the use of a blockmodel to extract context from features collected from mobile phone logs, and use a scoring system on the extracted features to identify the likely next location [14]. Finally, in [15], Gao et al. investigate the reverse of the standard problem. Instead of taking a future time and asking where the user will be, they take a location and ask when the user will next visit, achieved by modelling the probability distribution of a user visiting each location for all possible times of day using Bayesian inference, and returning the time calculated to be most probable from historical information.

3. Gradient-based Visit Extractor

Adopting the two-step approach for location extraction, this section presents the Gradient-based Visit Extractor (GVE), which extracts visits from geospatial trajectories and addresses the drawbacks of existing approaches, making it capable of extracting visits from a wider variety of datasets. Section 2.2 discussed existing approaches to visit extraction, with the STA extraction algorithm identified as the current state-of-the-art [6]. Although an improvement on previous techniques, this algorithm has several limitations. Firstly, the algorithm assumes that the data occurs in even time slices. While some domains have regular and predictable data recording, for the purpose of extracting visits from data collected from devices carried by users, it is likely that conditions such as signal loss or lack of battery power would result in periods of time with varying collection rates, and periods with missing data entirely, thus resulting in non-evenly timesliced data. Secondly, the STA extraction algorithm requires the buffer to be full before consideration of a visit can occur, both imposing a delay on points being considered, as a sliding window of points is required for the weighted averaging filter to operate, and specifying a minimum bound on visit duration. The result of this is that the minimum visit length to be extracted must be known a priori and used to select appropriate parameters.

First described in [25], and presented in expanded form here with an additional consideration for handling large periods of missing data not present in the previous version, the Gradient-based Visit Extractor (GVE) algorithm is shown as Algorithm 1. The remainder of this section introduces and discusses the algorithm. In the following sections we demonstrate the applicability of GVE to the task of visit extraction, an evaluation lacking in previous work. Specifically, the evaluation performs a thorough exploration of the parameter space of the GVE algorithm and analyses properties of the visits extracted, through metrics discussed in Section 4, with results presented and discussed in Section 5. The evaluation continues with a comparison with the STA extraction algorithm and, through visit clustering with DBSCAN, discusses properties of the clustered locations, based on visits extracted from both the GVE and STA extraction algorithms. DBSCAN is selected due to its relevance to the domain and ability to extract clusters of arbitrary shape without the number of clusters being known a priori.

The Gradient-based Visit Extractor extracts visits from temporally-annotated geospatial trajectories by continually building a visit until adding another point would cause the recent trend of motion to be consistently away from the visit already extracted. Although similar in idea to STA, GVE can consider visits without having a full buffer of points over which to analyse the trend of motion and allows for points collected at a varying rate. Additionally, GVE is able to consider points as they arrive, while the STA extraction algorithm imposes a delay due to its method of averaging results over a window.

The buffer over which the trend of motion of the user is considered has a maximum size of n_{points}, but this buffer does not need to be filled for a comparison to take place. Parameters α and β are used to define

Algorithm 1 Gradient-based Visit Extractor algorithm

```
n_points, α, β, t_max // Input parameters
visits ← [ ] // Empty array, to be filled with visits
visit ← [ p_0 ] // Array containing the first point in the dataset
buffer ← [ p_0 ] // Buffer over which to consider trend of motion

function Process(point)

    buffer.append(point)
    if buffer.length > n_points then
        buffer.shift
    end if

    // A visit has ended if there is a significant amount of missing data or the user is moving away
    if TimeBetween(visit.last, point) > t_max or MovingAway?(visit, buffer) then
        if TimeBetween(visit.first, visit.last) > 0 then
            visits.append(visit) // If the visit has some duration, store it
        end if
        visit ← [ point ]
        buffer ← [ point ]
    else
        visit.append(point) // If the visit hasn't ended, add the point to the visit
    end if

end function

// The user is moving away from a visit if the gradient of the buffer is greater than the threshold
function MovingAway?(visit, buffer)

    if Gradient(visit, buffer) > Threshold(α, β, buffer.length) then
        return True
    else
        return False
    end if

end function
```

a *threshold* function on the size of the buffer. If the buffer contains a small number of points, for example two points, and a third point were to arrive that is further away from the first point than the second, it could be an indication that the user is travelling away from the first point or it could be attributed to noise. This problem is mitigated by using a negative logarithmic function to ensure that the threshold for trend of motion is higher when there are fewer points in the buffer. Trend of motion is defined using a gradient that includes both spatial and temporal components, therefore allowing for the possibility of points of varying temporal distances. The *gradient* of buffer b is defined as:

$$\textsc{Gradient}(v,b) = \frac{l(b)\sum_{p\in b}(t(b,p)\times d(v,p)) - \sum_{p\in b} t(b,p)\sum_{p\in b} d(v,p)}{l(b)\sum_{p\in b} t(b,p)^2 - (\sum_{p\in b} t(b,p))^2}$$

where $l(b)$ is the current buffer size, $t(b,p)$ is the temporal distance between the first point in the buffer and the current point, p, in seconds, and $d(v,p)$ is the distance between point p and the centroid of the current visit, v, in metres. A gradient greater than the threshold indicates that the visit has ended:

$$\textsc{Threshold}(\alpha,\beta,\mathit{length}) = -\mathit{log}\left(\mathit{length}\times\frac{1}{\beta}\right)\times\alpha$$

With the gradient summarising the movement trend of the user relative to the visit, and a threshold function that returns a value dependent upon the number of points over which the gradient was calculated, we are now able to determine when a visit has ended: namely, if the gradient is greater than the threshold for a given set of points then the visit can be marked as having ended. This ensures resilience to noise by monitoring the movement trend over a set of points, but still allows for visits with few points.

In addition to trend of motion exceeding the threshold, a visit ends if the temporal distance between the point being considered and the immediately preceding point is greater than the parameter t_{max} (line 14 of Algorithm 1), which is a similar technique to that used by Montoliu and Gatica-Perez to detect periods of missing data [24]. Without this consideration, if data were to be missing for several hours or days, but by chance the first point recorded after this period is geographically close to the last point recorded, the gradient would be extremely low and is likely to be below the threshold. In this instance, the point would be considered as part of the previous visit, even though a significant period of time had elapsed and the user could easily have left the visit and simply returned to a location nearby. By introducing a maximum time between consecutive points for them to be part of the same visit, t_{max}, we mitigate this issue. Conversely, two consecutive points that fall on the same time instance (i.e. there is no time between them), could cause a visit of no duration to be extracted. This is prevented in line 15 of Algorithm 1, where visits are only stored if they have a positive duration. A visit that consists of a single point, or a visit that consists of multiple points that fall on the same time instance would therefore be considered noise.

While parameter selection is dependent upon the specific application and dataset, we impose one primary constraint, namely $\beta > n_{points}$. If we permitted β to be less than n_{points}, it is possible for the threshold function to return a negative value when the size of the buffer exceeds β. Using a negative threshold would allow a visit to be characterised as having ended when the gradient returns a negative value, where a negative gradient indicates that the user is moving towards the centre of the visit, rather than away from it. By requiring $\beta > n_{points}$ we ensure that any value returned by the threshold function is positive.

4. Experimental Setup

In order to demonstrate the applicability of the GVE algorithm to the problem of visit extraction, and to aid in parameter selection for future applications, we conduct a thorough analysis of the algorithm and resultant visits extracted. This analysis aims to guide application developers towards appropriate parameter choices, as well as demonstrating properties of the algorithm itself. Towards this goal, this section discusses the experimental methodology followed by presenting the algorithms and data selected for comparison, along with metrics through which analysis is conducted. The results of this exploration can be found in Section 5.

Table 1: Parameter sets for evaluation

GVE	α	≥ 0	0.05, 0.1, 0.15, 0.2, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 2.0
	β	> 1	3, 6, 9, 12, 18, 24, 30, 36, 46
	n_{points}	> 1	2, 5, 10, 15, ..., 35
	t_{max}	> 1	300, 600, 900, 1200, 2400, 3600, 7200, 9000, 13500, 18000
STAe	D_{thres}	> 0	0.1, 0.25, 0.5, 0.75, 1, 2.5, 5, 7.5, 10, 12.5, 15, 17.5, 20
	N_{buf}	> 1	4, 6, 8, ..., 32
	N_d	≥ 1	${}^{N_{buf}}/_2$
DBSCAN	*eps*	≥ 0	1, 10, 25, 50, 75, 100, 250
	minpts	≥ 0	0, 1, 2, 5

4.1. Algorithms

Section 2 provides a discussion on existing techniques for visit extraction and clustering. For the purpose of evaluation, we select the STA extraction algorithm [6] as a comparison to GVE. The STA extraction algorithm was chosen as it is the current state-of-the art, offering significantly improved resilience to noise than existing approaches. For visit clustering, i.e. clustering extracted visits into locations, we select DBSCAN. While there are several approaches that have shown promise for this purpose, DBSCAN is widely used and has desirable properties, such as not requiring the number of locations to be known a priori [1, 5, 12, 24].

In order to ensure consistency, all implementations of the algorithms assume that distances are measured in metres, and time in seconds. Furthermore, we define the shape of a location to be the convex hull of the centroids of all visits belonging to the location. Summarising visits in such a way is an important part of the two-step approach to location extraction as it vastly reduces the complexity of the clustering procedure, and thus is a technique we employ here [1–3, 7, 24, 28].

4.2. Evaluation Parameters

With algorithms for evaluation selected, we narrow down the range of parameters considered. While it is not possible to consider all combinations of parameters, as many are unbounded, we select a range of parameters for each algorithm that aims to cover the parameter space as thoroughly as possible, ensuring that the results obtained are representative. The ranges of values selected for each parameter can be found in Table 1, and this section briefly discusses the parameters for each algorithm.

GVE takes 4 parameters: α and β alter the threshold function, where a gradient above the threshold indicates the end of a visit, with α scaling in the y-dimension, and β in the x-dimension. The n_{points} parameter specifies the maximum size of the buffer, and t_{max} specifies the maximum amount of time between two consecutive points permitted for the points to belong to the same visit. Only parameter combinations that obey the constraint $\beta > n_{points}$ are considered.

STA extraction algorithm has a fixed size buffer of length N_{buf}, and uses a weighted averaging function of window size N_d to characterise the trend of motion, with D_{thres} (measured in metres) specifying the threshold for the averaged buffers, above which the visit is considered to have ended. A relationship between N_d and N_{buf} was proposed by the authors of the STA extraction algorithm, namely $N_d = \frac{N_{buf}}{2}$, and we adopt this approach here [6].

DBSCAN is a density-based clustering algorithm, whose two parameters, *minpts* and *eps* (ϵ), specify the required density of a cluster, with *minpts* specifying the minimum number of points required within *eps* metres for a cluster to be extracted. A detailed explanation of how these parameters affect the extracted clusters can be found in [12].

With visit extraction, it is feasible to explore the parameter space by performing extraction using all possible combinations of the parameters shown in Table 1 due to the linear time requirement of the algorithms. However, performing visit clustering on all sets of extracted visits, and with all combinations of parameters, becomes infeasible due to the increased run-time of DBSCAN. Instead, we use the exploration of the parameter space of the visit extraction algorithms to select 2 appropriate and representative sets of visits for each algorithm and use these to conduct an exploration of the parameter space for visit clustering, shown later in Section 5.2. These sets of visits are empirically selected to provide representative trends with regards to the relationship between the parameter space of the clustering algorithm and the metrics used to characterise them.

4.3. Metrics

Selecting appropriate parameters for visit extraction and clustering is a difficult problem, not least because of the limited knowledge available relating to how altering parameters would affect the properties of the visits and locations produced. Adopting two application scenarios, namely the use-cases presented in Section 1.2, we explore the properties of visits and locations to gain a greater understanding of the applicability of certain algorithms and parameter sets.

This exploration is conducted with respect to metrics including the duration of visits extracted, proportion of data points classed as noise (as noise points indicate movement, this proportion is analogous to proportion of time spent travelling between locations), and the area of locations extracted. In these cases, while an application developer may not have specific target values, it is feasible that an acceptable range of values could be determined. As an example, if an application requires room-size locations, then average location area would be expected to be small, or if data were to be collected from a vehicle that is in motion for most of the day, noise proportion would be expected to be high.

4.3.1. Ground-truth Comparisons

Clustering is generally viewed as an unsupervised learning problem. In order to investigate how the results of clustering compare to an expected outcome, a ground-truth of visited locations needs to be available. If such a ground-truth exists, comparisons to the locations determined by the algorithms can be performed using the set-similarity measure *Dice's coefficient*:

$$QS = \frac{2|A \cap B|}{|A| + |B|}$$

where QS is the quotient of similarity in range $[0, 1]$. A value of 1 indicates that the sets (A and B) are identical, while 0 indicates no overlap. Although Dice's coefficient is designed for comparing the contents of two sets, it can be used over clusters by considering $|A \cap B|$ to represent the area covered by both clusters, and $|A|+|B|$ to be the sum of the areas of the clusters, effectively providing a score for the *location properties* use-case.

4.4. Evaluation Data

Our primary evaluation data comes from the Nokia Mobile Data Challenge (MDC) Dataset [19, 20], which is the largest dataset of its kind available for research purposes and consists of high-accuracy location and usage data collected from smartphones carried by 191 users over a period of 2 years. From this dataset we select the GPS location data from 9 MDC users with different lengths of data, ensuring resultant users have a wide range of point cardinalities. In the process of protecting the privacy of users who contributed data to the MDC Dataset, the creators opted to identify regions in which users live and work and truncate the recorded latitude and longitude values for any points recorded in these areas, lowering accuracy and removing all variance throughout these time periods. For our analysis we opt to exclude these points, and treat them as missing data. The algorithms that we are discussing assume variance in data and, therefore, testing them on datasets with this artificial property may produce unexpected results. On the other hand, the expectation of periods of missing data is natural for all geospatial datasets recorded through devices carried by the user, so no loss of generality should occur by treating such data as missing. A second caveat of the MDC dataset is that the licensing terms of the data prohibit the distribution of maps showing user data. In order to provide representations of extracted visits and locations, we employ a second dataset consisting of points collected from the smartphone belonging to the first author of this paper in a manner designed to replicate the collection methodology of the MDC dataset, spanning 114 days. Although only containing data from a single user, all trends discussed in Section 5 relate equally to the MDC dataset and this additional data.

4.4.1. Ground Truth Comparison Locations

The selection of distinct locations to use for comparison is extremely difficult in the absence of a ground truth provided by the users themselves. The MDC Dataset includes user-contributed *places* which are identified by algorithm and then labelled by the user. Unfortunately, this set is incomplete as it includes places for only a small proportion of the users in the dataset. Furthermore, manually overlaying the identified places on satellite imagery of the area provided no clear identification of the building or geographic feature to which the place belonged, as the corresponding points covered several buildings or landmarks. Instead of relying on these self-labelled places, we opt to identify manually 5 buildings or distinct geographic regions (e.g. a park) that were unambiguously visited for each user throughout the course of the data. This is achieved by overlaying all points belonging to a user on satellite imagery of the area, using Google Earth, and inspecting the resulting plots to identify dense areas of points that had a clear centre on a geographic feature or building. For each of the 9 users in our evaluation, we select 5 locations and summarise each location into a minimum set of points, which is stored for later comparison. Although this methodology is not flawless, as it may include locations visited a limited number of times, it provides an indication of how well the algorithms perform at extracting the geographical shape of visited locations as described in the *location properties* use-case in Section 1.2.

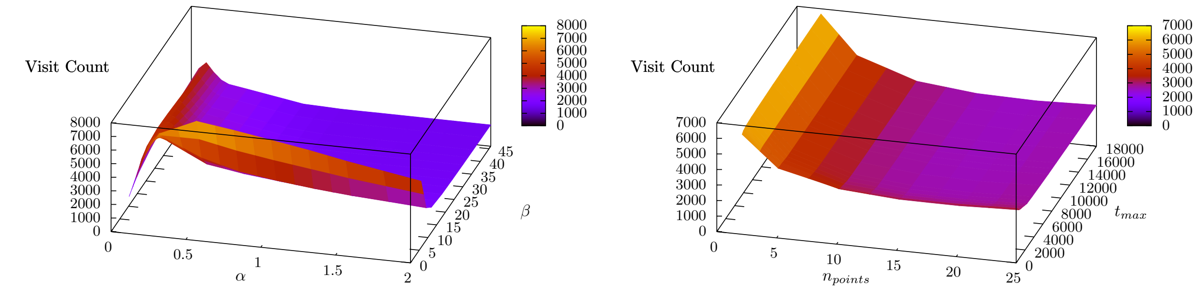
5. Parameter Exploration

Towards our goals of demonstrating the applicability of GVE to the task of visit extraction, and aiding in parameter selection for applications of the algorithm, this section presents the results observed from visit extraction under the methodology proposed in Section 4. This section begins with a parameter exploration of GVE, with results presented in Section 5.1, by analysing properties of the visits extracted with respect to the metrics discussed previously, then goes on to evaluate the locations extracted by clustering visits from both GVE and the STA extractor using DBSCAN (Sec. 5.2). Finally, Section 5.3 presents an evaluation of the applicability of both GVE and the STA extraction algorithm to the use-cases originally discussed in Section 1.2, when locations are determined by DBSCAN.

5.1. Gradient-based Visit Extractor (Visit Extraction)

The effect of the parameters on the metrics discussed in Section 4.3 can be seen in Figures 1-3. In each figure, surface plots are shown with separate plots for parameters α and β and for n_{points} and t_{max}, requiring the two missing parameters to be constrained. For supervised learning problems, it is possible that there would be optimal values that can be selected to achieve this goal. However, no such values exist for unsupervised clustering, and so we opt for constraining parameters such that the resultant plots provide a representative view of the unconstrained parameters. Specifically, we opt for $\alpha = 0.1$, $\beta = 30$, $n_{points} = 5$, $t_{max} = 3600$, which were empirically determined to provide a representative view of the trends present in the relationships between the parameter space and the metrics. It is important to note that these graphs are presented to show the trends, rather than specific values extracted. The specific values (e.g. number of visits extracted) vary considerably based on the dataset provided to the algorithm, while the trends remain constant.

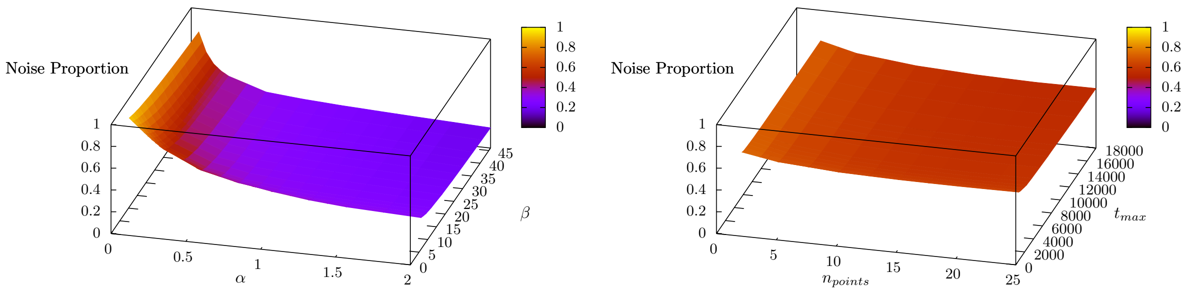
Figure 1 shows the number of visits extracted for different GVE parameters, demonstrating their effects. Specifically, Figure 1a shows the parameters α and β, which control the *threshold* function (a function on the current number of points in the buffer). The number of visits extracted increases and their duration becomes shorter when either α or β are lowered. When α becomes extremely small (< 0.25), this trend changes abruptly. As the average number of data points per visit decreases, a significant proportion of these visits contain only a single point. In the context of visit extraction, visits consisting of no duration, including those that contain a single point, are considered noise and disregarded; hence the number of visits extracted



(a) $n_{points} = 5, t_{max} = 3600$

(b) $\alpha = 0.1, \beta = 30$

Figure 1: Parameter effect on the number of visits extracted



(a) $n_{points} = 5, t_{max} = 3600$

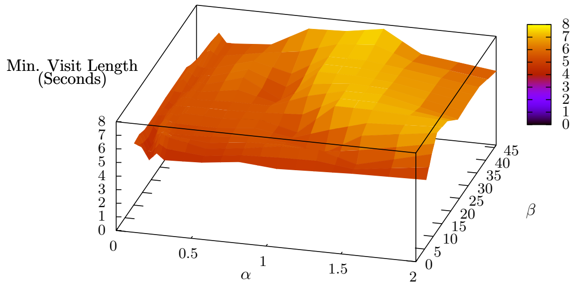
(b) $\alpha = 0.1, \beta = 30$

Figure 2: Parameter effect on the proportion of points designated as noise

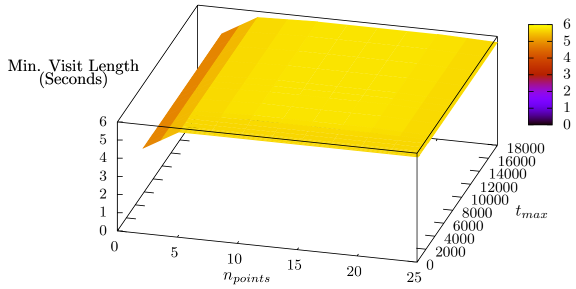
reduces, while the proportion of noise increases drastically (Figure 2a). The number of visits extracted by the algorithm has a direct impact on the *accurate visits* use-case, in that the correct characterisation of where a user spends their time requires the correct number of visits. If too few visits are extracted, multiple real-world visits will have been merged, causing a loss of information, thus invalidating the utility afforded by the start and end times of each visit.

When considering the number of visits extracted, the maximum time between consecutive points permitted for them to belong to the same visit, t_{max}, has limited impact as it affects very few visits (Figure 1b). With small t_{max} values, visits are split more frequently as smaller amounts of missing data cause a visit to be ended, thereby increasing the number of visits extracted. With larger values, the impact is no longer noticeable due to the relatively small number of periods of missing data in the dataset. However, setting t_{max} too large causes larger periods of missing data to be included within visits where it may have been possible for a user to leave the visit and return, reducing the accuracy of visit properties. Similarly, n_{points} is most impactful for small values. With a small n_{points} value, the gradient is calculated only over a small number of samples and is therefore more susceptible to noise, prematurely ending visits and causing the number of visits extracted to increase.

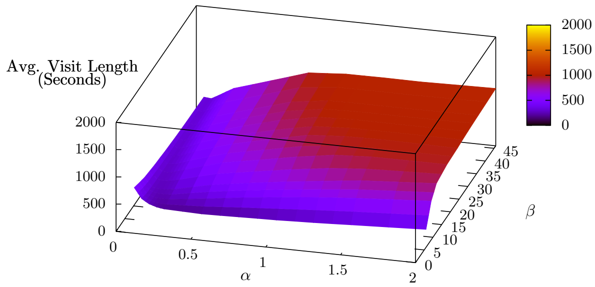
The relationships between the parameters and the number of visits extracted are corroborated by Figures 3c and 3d, which show that when the number of visits extracted decreases, the average length of the extracted visits increases. While the maximum visit length (Figures 3e and 3f) also follows this general trend, it is important to note that low values of α do not have the same effect as with average visit length because the behaviour only affects very short visits.



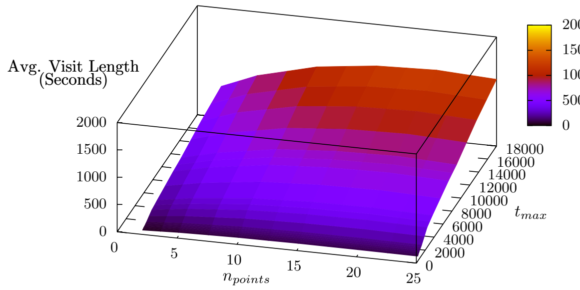
(a) Min ($n_{points} = 5, t_{max} = 3600$)



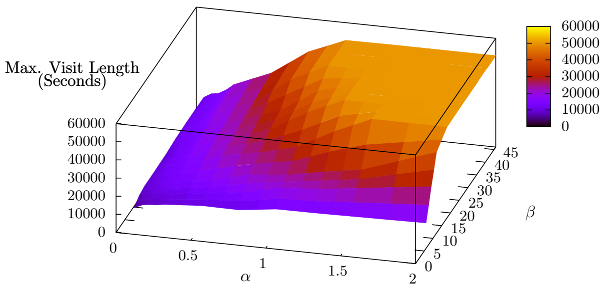
(b) Min ($\alpha = 0.1, \beta = 30$)



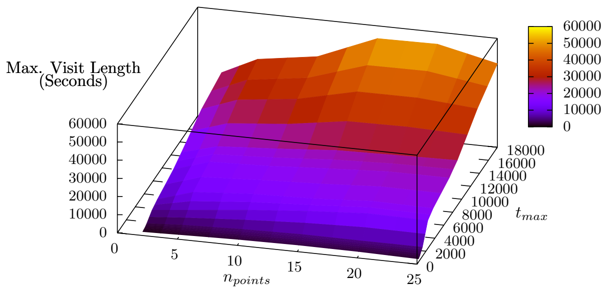
(c) Avg ($n_{points} = 5, t_{max} = 3600$)



(d) Avg ($\alpha = 0.1, \beta = 30$)



(e) Max ($n_{points} = 5, t_{max} = 3600$)



(f) Max ($\alpha = 0.1, \beta = 30$)

Figure 3: Parameter effect on the length of visit extracted (seconds)

Table 2: Parameter sets considered for GVE algorithm

ID	α	β	n_{points}	t_{max}	Average Visit Count
GVE-1	0.25	12	10	1200	4914
GVE-2	1.5	36	30	13500	1293

Table 3: Parameter sets considered for STA extraction algorithm

ID	D_{thres}	N_{buf}	N_d	Average Visit Count
STAe-1	7.5	4	2	1319
STAe-2	15	26	13	126

While the relationships between number of visits, average and maximum visit duration and proportion of noise are heavily correlated, minimum visit duration (Figures 3a and 3b) does not follow this trend. In fact, the length of the minimum visit extracted is not heavily dependent upon any parameter since GVE does not impose a minimum bound on visit duration. While parameters may still influence the length of the shortest visit, with a large and varied dataset the effect of the parameters on minimum visit length is minimal. The results show that the shortest visit is extracted when both α and β are low, as this increases the likeliness of a visit being marked as having ended sooner. Additionally, lower values of n_{points} cause the length of the shortest visit to decrease, as the algorithm is only considering the trend of motion over a small number of points. Not imposing a minimum bound on duration is beneficial because it allows arbitrarily short visits to be extracted, although in some cases this may not be desirable. If the dataset were collected at a very high rate, extremely short visits could be extracted. In certain applications these may not be required and so can be removed easily in a single step once visit clustering has completed by classifying the points belonging to them as noise retroactively.

5.2. DBSCAN (Visit Clustering)

With the properties of visits extracted through GVE better understood, the task now becomes clustering visits together to form locations. As detailed in Section 4.2, visit clustering is performed on sets of visits extracted using parameters selected to provide a representative overview of the relationship between the parameter space and characterisation metrics employed. Parameters have been selected for both the GVE and STA extraction algorithms for this purpose, and presented in Tables 2 and 3 respectively. Although a detailed analysis of the parameter space of the STA extraction algorithm is omitted for brevity, the number of visits extracted for all combinations of parameters was significantly lower than with GVE, making selection of sets of parameters that produced similar numbers of visits infeasible without constraining GVE's parameters too heavily, and so only one set of parameters was selected for each algorithm that produced a similar number of visits (GVE-2 and STAe-1). Clustering was then performed on these extracted visits by DBSCAN with the parameters outlined in Section 4.2.

Figure 4 shows how the number of locations extracted varies with different algorithms and sets of parameters for DBSCAN, for two different values of *minpts* (1 and 5). The effect of the *minpts* parameter for DBSCAN is consistent and the remaining values not shown are omitted for brevity. The results show that DBSCAN performs predictably, and that the largest number of locations are extracted with the smallest *eps* and *minpts* parameters (i.e. requiring the lowest density of points to consider a location). Increasing *minpts* would require a higher density, thus creating fewer locations, and increasing *eps* would increase the area to consider for a location, thereby creating fewer, but larger, locations. This finding is reinforced by Figure 5, showing the average size of locations and the standard deviation of location sizes. The appropriate selection of parameters for clustering is important for both use-cases presented in Section 1.2. For *accurate visits*, locations are used to group visits together so that visits to the same location can be identified; therefore, locations that are too large or small will cause visits that are not to the same location to be grouped together incorrectly. The clustering step has a greater impact on *location properties*, however, as this use-case is solely interested in the clustered locations. Setting the parameters such that the locations extracted are too small

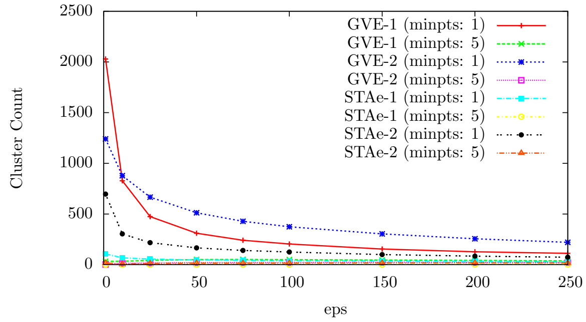
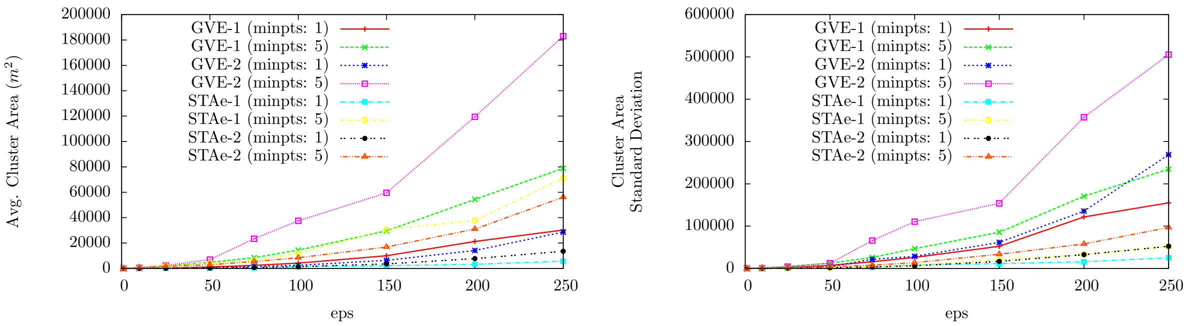


Figure 4: Parameter effect on the number of locations extracted for DBSCAN



(a) Area (m^2)

(b) Standard deviation

Figure 5: Parameter effect on the average size of locations for DBSCAN

will cause erroneous locations to be found as real-world locations are split into multiple parts, while if the extracted locations are too large, real-world locations will be merged.

The relationship between the parameter spaces and the similarity to the ground truth cluster set (calculated using Dice's coefficient, and representing a score of correctness for our *location properties* use-case) is less straightforward. Figure 6 shows the relationship when calculating Dice's coefficient for the location with the highest similarity to each ground truth location. As *eps* is increased, the similarity begins to increase and then slowly falls back down as the value is increased further. Dice's coefficient is maximal when the location and ground truth clusters are identical. If the extracted location is either larger or smaller than the ground truth, the similarity is lower. The results in Figure 6 show that, of the 4 sets of visits used as a basis for location clustering, GVE-1 (Table 2) produces locations that are closest to the ground truth.

5.3. Relation to Use-cases

Demonstrating the applicability of GVE to the problem of visit extraction, we focus on the two use-cases presented in Section 1.2. The first, *accurate visits*, is concerned with characterising where a user spent their time, achieved through the accurate identification of visits and subsequent accurate grouping together of visits to form locations. While it is not possible to state categorically the optimum parameters for achieving this goal for a given set of trajectories in absence of a concrete ground truth, we can use the exploration of the parameter space to reason about suitable ranges of parameters. Using knowledge about the source and length of trajectories, it would be reasonable to suggest an expected minimum number of visits, thereby allowing us to constrain the available parameter space. Furthermore, the expected ratio of time that a user spent travelling against time spent at a visit could be estimated, imposing minimum values for the proportion of noise in the trajectories, as the proportion of noise is approximately equivalent to proportion of time spent travelling between visits.

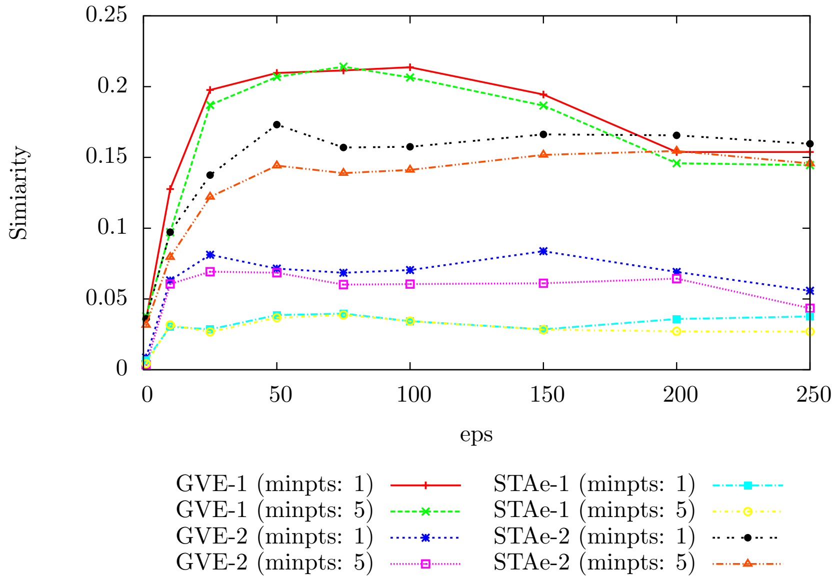


Figure 6: Parameter effect on the similarity to ground truth for DBSCAN

5.3.1. Accurate Visits

In order to consider the applicability of each algorithm to the *accurate visits* use-case, we investigate the properties of visits that are produced when using both algorithms. To achieve this, we use knowledge about the data to make certain assumptions about properties of visits that we expect to see:

- Users make an average of between 2 and 15 visits per day;
- The maximum length of a single visit does not exceed 2 days (172800 seconds);
- Each user made at least one short visit, for example to buy coffee or pay for a service, and so the minimum visit duration must be less than 10 minutes (600 seconds);
- Each user spent at least 60% of their time at a visit location, and no more than 40% travelling between them (a noise proportion of less than 0.4).

While it is true that these assumptions may not hold for every individual, for example it may be possible for a delivery driver to average more than 9.6 hours per day travelling (i.e. 40% of 24 hours), we believe that they apply to the majority of people and, specifically, the majority of users who provided data for the MDC dataset. However, the assumption of maximum visit length is not as likely to hold with certain users if, for example, a hospital stay required them to remain stationary for more than 48 hours, or missing data occurred in the dataset and this was recorded as a visit by the STA extraction algorithm (GVE uses a parameter, t_{max}, to overcome this problem). We therefore investigate the effect of this set of assumptions with and without the assumption of maximum visit length. Using all sets of visits extracted with the parameters discussed in Section 4.2, we select only those that conform to these assumptions, with results shown in Tables 4 and 5 (with and without the maximum visit length assumption, respectively), listed per user, for both MDC users (represented by an ID number between 1 and 9) and the first author of this paper (represented by A). In both tables, the column titled *Matching* shows the percentage of parameter sets that produce visits consistent with the assumptions. It is important to note here that there are considerably more possible sets of parameters for GVE than the STA extraction algorithm (4290 and 195, respectively) as the GVE algorithm takes several more parameters and all combinations of parameters are considered. Despite there being more runs for GVE, both sets of parameters were selected to cover the parameter space

Table 4: Visit extraction summary (with maximum visit length assumption)

Algorithm	User	Days	% Matching	Visits	Avg. Length (S)	Noise Proportion
GVE	1	6	32.005	53 - 90	433 - 4018	0.143 - 0.398
	2	497	42.541	2816 - 7454	388 - 3748	0.054 - 0.272
	3	548	39.744	2326 - 6438	89 - 1464	0.177 - 0.4
	4	550	67.366	1223 - 8246	106 - 6042	0.032 - 0.4
	5	144	66.503	469 - 2156	138 - 4938	0.048 - 0.4
	6	297	0.023	947 - 947	1405 - 1405	0.395 - 0.395
	7	178	48.368	763 - 2670	67 - 1869	0.153 - 0.4
	8	187	54.149	753 - 2804	179 - 5084	0.054 - 0.399
	9	23	8.462	165 - 345	137 - 2637	0.232 - 0.4
	A	114	59.557	326 - 1634	3005 - 17016	0.01 - 0.09
STA extraction	1	6	8.205	21 - 55	5223 - 12046	0.328 - 0.398
	2	497	0.0	-	-	-
	3	548	0.0	-	-	-
	4	550	0.0	-	-	-
	5	144	4.103	291 - 709	1652 - 4364	0.24 - 0.371
	6	297	0.0	-	-	-
	7	178	0.0	-	-	-
	8	187	0.0	-	-	-
	9	23	0.0	-	-	-
	A	114	16.923	299 - 1431	2565 - 17380	0.069 - 0.34

as well as possible, and so the difference in the number of parameter sets should not have an impact on the results presented.

Focusing on the results for the GVE algorithm, in all cases at least one set of parameters produces results consistent with our assumptions. User 6 is an anomaly, however, as only a single set of parameters produced such a result, indicating that the assumptions may not be valid for this user. In this specific case, the applicable parameter sets are restricted by the assumption placed on noise proportion, indicating that the user may spend more than 40% of their time travelling. Considering the STA extraction algorithm, the number of users for whom visits are extracted consistent with expectations is significantly reduced (3 out of 10 in Table 4, rising to 6 out of 10 when relaxing the maximum visit length assumption, shown in Table 5). Under these assumptions, therefore, it appears that GVE produces visits that are likely to be applicable to the *accurate visits* use-case.

As discussed in Section 4.4, the licence for the MDC data prohibits the distribution of maps that may cause locations or users to be identified. For illustration, we therefore plot sample visits using trajectories from the first author of this paper, where the visits are from parameters that produce visit sets consistent with the assumptions outlined previously. Specifically, we select parameters by observing the results of plotting visits on a map and selecting only parameters that create visits consistent with the expectation that visits should not span multiple buildings. From this, the result with the lowest proportion of points designated as noise is selected for each algorithm. With visual observation confirming the correctness of visits, the visits with the lowest proportion of noise provide characterisation for the largest proportion of time. The plots are shown in Figure 7 and the properties of the visits listed in Table 6. In Figure 7, visits are represented by different colours with the convex hull of the points belonging to each visit illustrated in the same colour as the points. The convex hulls have been included for clarity so as to present unambiguously which points belong to which visit. In these specific cases, GVE has extracted several more, and shorter, visits than STA and also covers more of the trajectories, as demonstrated by a lower noise proportion, further indicating its applicability to the problem of visit extraction.

Table 5: Visit extraction summary (without maximum visit length assumption)

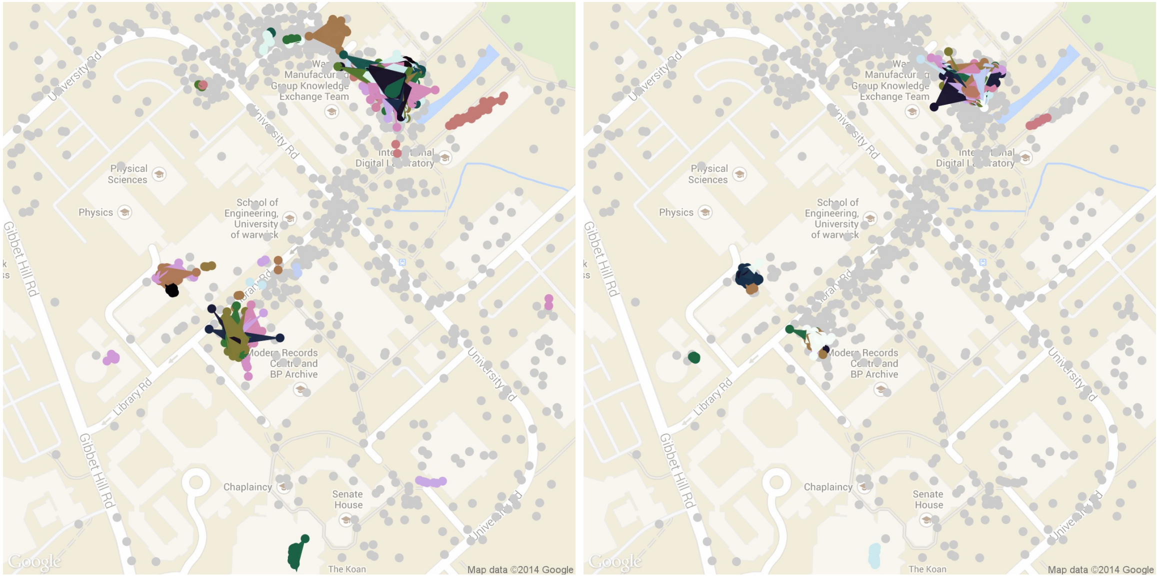
Algorithm	User	Days	% Matching	Visits	Avg. Length (S)	Noise Proportion
GVE	1	6	32.005	53 - 90	433 - 4018	0.143 - 0.398
	2	497	51.375	2436 - 7454	388 - 5441	0.053 - 0.272
	3	548	39.744	2326 - 6438	89 - 1464	0.177 - 0.4
	4	550	67.366	1223 - 8246	106 - 6042	0.032 - 0.4
	5	144	66.503	469 - 2156	138 - 4938	0.048 - 0.4
	6	297	0.023	947 - 947	1405 - 1405	0.395 - 0.395
	7	178	48.368	763 - 2670	67 - 1869	0.153 - 0.4
	8	187	54.149	753 - 2804	179 - 5084	0.054 - 0.399
	9	23	8.462	165 - 345	137 - 2637	0.232 - 0.4
	A	114	90.396	228 - 1634	3005 - 24835	0.007 - 0.09
STA extraction	1	6	8.205	21 - 55	5223 - 12046	0.328 - 0.398
	2	497	11.795	1028 - 6707	996 - 16262	0.179 - 0.399
	3	548	0.0	-	-	-
	4	550	3.077	1128 - 2185	3303 - 8853	0.246 - 0.361
	5	144	4.103	291 - 709	1652 - 4364	0.24 - 0.371
	6	297	0.0	-	-	-
	7	178	0.0	-	-	-
	8	187	10.256	400 - 1359	2835 - 12827	0.206 - 0.384
	9	23	0.0	-	-	-
	A	114	16.923	299 - 1431	2565 - 17380	0.069 - 0.34

Table 6: Metric summary of visualised visits

	Visits	Avg. Visit (s)	Noise Proportion
GVE	695	7377	0.063
STA extraction	558	8991	0.118

5.3.2. Location Properties

With the second use-case, *location properties*, the aim is to extract locations that correctly represent the geographical shape of buildings and locations known to have been visited by the users. Again, identification of optimal parameters for this case requires a concrete ground truth. However, unlike the *accurate visits* use-case, the manual identification and creation of a partial ground truth can be performed by observation. While creating a ground truth for visits requires analysing each individual point to determine whether it could be seen as starting or ending the visit, identifying locations visited can be performed top-down by observing dense groups of points that fall on geographical landmarks such as buildings. With a set of observed locations (Section 4.4.1) and a similarity measure (Dice's coefficient), the selection of appropriate parameters can be performed through a greedy search that aims to maximise this similarity value. Figure 8 shows the identified ground truth locations and Figure 9 shows the clusters extracted that maximise Dice's coefficient relative to the extracted locations for both visit extraction algorithms, with a summary of parameters and their corresponding Dice's coefficient presented in Table 7. The visit extraction parameters used were the same as in Figure 7. The results here demonstrate that GVE is capable of extracting visits that provide a foundation for location extraction through clustering as all buildings were identified. Determining the exact shapes of buildings is likely to be infeasible due to the inaccuracies inherent in location detection methods, the summary of visits into a single point, and the fact that the individual may not have visited all areas of the building. Despite this, the buildings themselves were correctly identified as locations meaningful to the user, with a greater similarity to the ground-truth than the STA extraction algorithm produced.



(a) GVE ($\alpha = 0.05$, $\beta = 36$, $n_{points} = 5$, $t_{max} = 1200$) (b) STA extraction ($D_{thres} = 2.5$, $N_{buf} = 6$, $N_d = 3$)

Figure 7: Extracted visits

Table 7: Parameter summary of visualised clusters

Visit Clusterer	Dice's Coefficient	minpts	eps
GVE	0.28	2	25
STA extraction	0.23	5	75

6. Conclusion

This paper has considered location extraction from geospatial trajectories as a two-step process. In the first step, periods of low mobility, referred to as *visits*, are extracted from the data, with these visits then clustered into locations. Performing visit extraction as a precursor step, in such a manner, allows the summarisation of a large dataset into relatively few visits, drastically reducing the complexity of clustering. Once extracted, locations such as these are used as the basis for many applications and services, including location prediction and systems that utilise context in order to reason about users as both individuals and as groups.

The primary focus of this paper has been the presentation and analysis of GVE, the Gradient-based Visit Extractor algorithm, that extracts periods of low mobility from geospatial data while maintaining resilience to noise and overcoming the drawbacks of existing techniques. Specifically, GVE does not place a minimum bound on visit duration, has no assumption of evenly spaced observations and does not impose a delay on trajectory points being considered, making it amenable to visit extraction in real-time from a larger variety of data sources than existing approaches. In addition to presenting the algorithm, this paper has provided a comprehensive analysis of the properties of the visits extracted through an exploration of the parameter space, thus providing application developers with knowledge to aid in parameter selection. The applicability of GVE to the task of visit extraction was then demonstrated through both a comparison with the STA extraction algorithm, the current state-of-the-art, and analysis of locations determined through clustering with DBSCAN. In both of these investigations positive results demonstrating the suitability of GVE were achieved, with evidence indicating increased accuracy over existing approaches. This quantitative evaluation, lacking from previous work, demonstrates the applicability of the Gradient-based Visit Extractor (GVE) algorithm to the task of visit extraction and, consequently, as a precursor step to location extraction from geospatial trajectories.

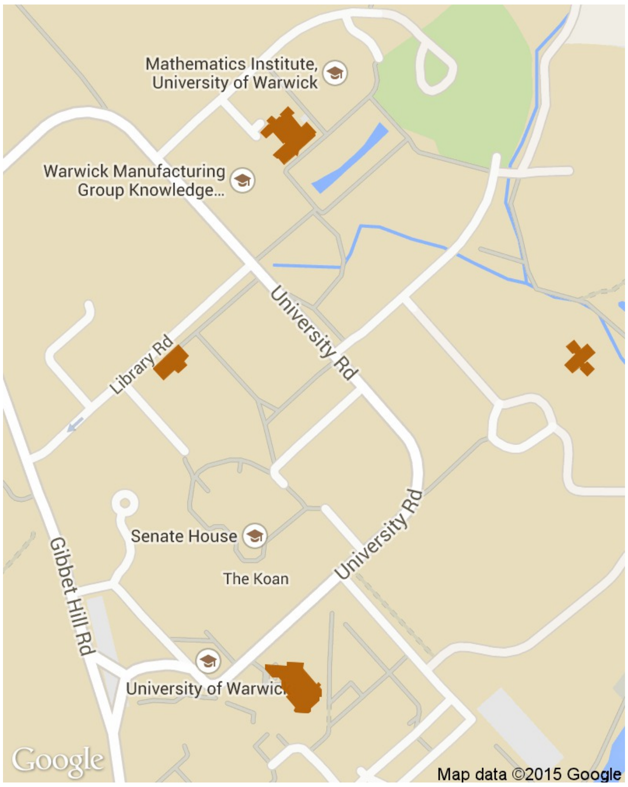
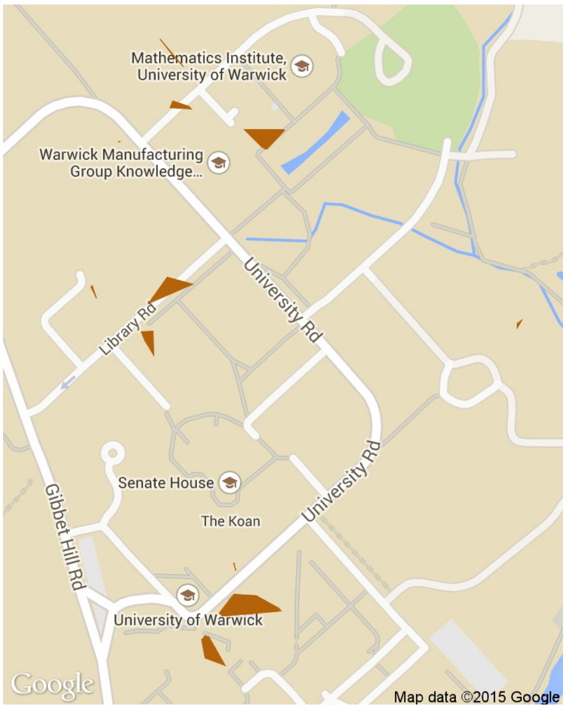
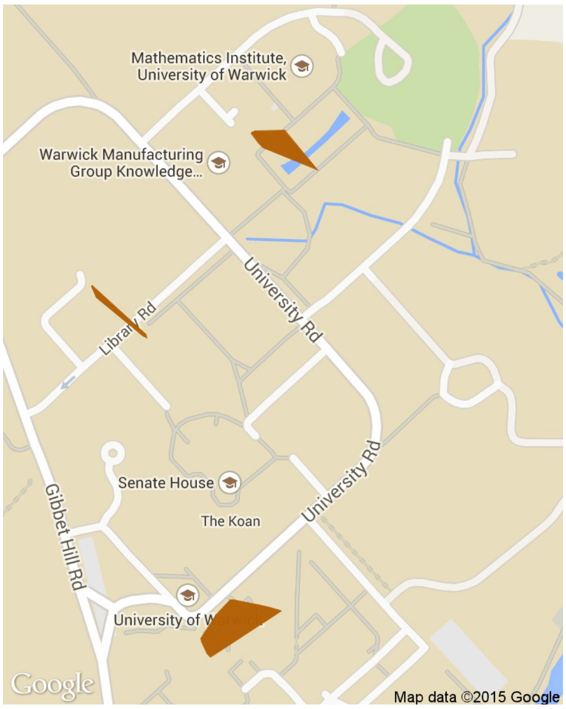


Figure 8: Ground truth clusters



(a) GVE - DBSCAN



(b) STAe - DBSCAN

Figure 9: Extracted clusters maximising Dice's coefficient

Acknowledgements

Portions of the research in this paper used the MDC Database made available by Idiap Research Institute, Switzerland and owned by Nokia [19, 20].

References

[1] Gennady Andrienko, Natalia Andrienko, Georg Fuchs, Ana-Maria Olteanu Raimond, Juergen Symanzik, and Cezary Ziemlicki. Extracting Semantics of Individual Places from Movement Data by Analyzing Temporal Patterns of Visits. In *Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 5–8, Orlando, 2013.

[2] Gennady Andrienko, Natalia Andrienko, Christophe Hurter, Salvatore Rinzivillo, and Stefan Wrobel. From Movement Tracks Through Events to Places: Extracting and Characterizing Significant Places from Mobility Data. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology*, pages 161–170, Providence, 2011.

[3] Daniel Ashbrook and Thad Starner. Learning Significant Locations and Predicting User Movement with GPS. In *Proceedings of the 6th International Symposium on Wearable Computers*, pages 101–108, Seattle, 2002.

[4] Daniel Ashbrook and Thad Starner. Using GPS to Learn Significant Locations and Predict Movement Across Multiple Users. *Personal and Ubiquitous Computing*, 7(5):275–286, 2003.

[5] Roland Assam and Thomas Seidl. Context-based Location Clustering and Prediction using Conditional Random Fields. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia*, pages 1–10, Melbourne, 2014.

[6] Athanasios Bamis and Andreas Savvides. Lightweight Extraction of Frequent Spatio-Temporal Activities from GPS Traces. In *Proceedings of the 31st Real-Time Systems Symposium*, pages 281–291, San Diego, 2010.

[7] Athanasios Bamis and Andreas Savvides. Exploiting Human State Information to Improve GPS Sampling. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 32–37, Seattle, 2011.

[8] Derya Birant and Alp Kut. ST-DBSCAN: An Algorithm for Clustering Spatial–Temporal Data. *Data & Knowledge Engineering*, 60(1):208–221, 2007.

[9] Zhenyu Chen, Yiqiang Chen, Shuangquan Wang, and Zhongtang Zhao. A Supervised Learning Based Semantic Location Extraction Method Using Mobile Phone Data. In *Proceeding of the IEEE International Conference on Computer Science and Automation Engineering*, pages 548–551, Zhangjiajie, 2012.

[10] Yohan Chon, Elmurod Talipov, Hyojeong Shin, and Hojung Cha. Mobility Prediction-based Smartphone Energy Optimization for Everyday Location Monitoring. In *Proceedings of the 17th International Conference on World Wide Web*, pages 82–85, Seattle, 2011.

[11] Trinh Minh Tri Do and Daniel Gatica-Perez. The Places of Our Lives: Visiting Patterns and Automatic Labeling from Longitudinal Smartphone Data. *IEEE Transactions on Mobile Computing*, 13(3):638–648, 2013.

[12] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining*, pages 226–231, Portland, 1996.

[13] Katayoun Farrahi and Daniel Gatica-Perez. Learning and Predicting Multimodal Daily Life Patterns from Cell Phones. In *Proceedings of the International Conference on Multimodal Interfaces*, pages 277–280, Cambridge, 2009.

[14] Jun Fukano, Tomohiro Mashita, Takahiro Hara, Kiyoshi Kiyokawa, Haruo Takemura, and S Nishio. A Next Location Prediction Method for Smartphones Using Blockmodels. In *Proceedings of the IEEE Conference on Virtual Reality*, pages 1–4, 2013.

[15] Huiji Gao, Jiliang Tang, and Huan Liu. Mobile Location Prediction in Spatio-Temporal Context. In *Proceedings of the Nokia Mobile Data Challenge (MDC) Workshop in Conjunction with Pervasive*, Newcastle, 2012.

[16] Yu Gong, Yong Li, Depeng Jin, Li Su, and Lieguang Zeng. A Location Prediction Scheme Based on Social Correlation. In *Proceedings of the IEEE Vehicular Technology Conference*, pages 1–5, 2011.

[17] Ramaswamy Hariharan and Kentaro Toyama. Project Lachesis: Parsing and Modeling Location Histories. In *Proceedings of the 3rd International Conference on Geographic Information Science*, pages 106–124, Adelphi, 2004.

[18] Jong Hee Kang, William Welbourne, Benjamin Stewart, and Gaetano Borriello. Extracting Places from Traces of Locations. In *Proceedings of the 2nd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots*, pages 110–118, Philadelphia, 2004.

[19] Niko Kiukkonen, Jan Blom, Olivier Dousse, Daniel Gatica-Perez, and Juha Laurila. Towards Rich Mobile Phone Datasets: Lausanne Data Collection Campaign. In *Proceedings of the First Workshop on Modeling and Retrieval of Context*, Berlin, 2010.

[20] Juha K Laurila, Daniel Gatica-Perez, Imad Aad, Jan Blom, Olivier Bornet, Trinh Minh Tri Do, Olivier Dousse, Julien Eberle, and Markus Miettinen. The Mobile Data Challenge: Big Data for Mobile Computing Research. In *Proceedings of the Nokia Mobile Data Challenge (MDC) Workshop in Conjunction with Pervasive*, Newcastle, 2012.

[21] Quannan Li, Yu Zheng, Xing Xie, Yukun Chen, Wenyu Liu, and Wei-Ying Ma. Mining User Similarity Based on Location History. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 34:1–34:10, Irvine, 2008.

[22] Lin Liao, Donald J Patterson, Dieter Fox, and Henry Kautz. Learning and Inferring Transportation Routines. *Artificial Intelligence*, 171(5-6):311–331, 2007.

[23] James MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the 5th Berkeley Symposium on Math, Statistics, and Probability*, pages 281–297, Berkeley, 1967.

[24] Raul Montoliu and Daniel Gatica-Perez. Discovering Human Places of Interest from Multimodal Mobile Phone Data. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia*, pages 12:1–12:10, Limassol, 2010.

[25] Alasdair Thomason, Nathan Griffiths, and Matthew Leeke. Extracting Meaningful User Locations from Temporally Annotated Geospatial Data. In *Internet of Things: IoT Infrastructures*, volume 151 of *LNICST*, pages 84–90. Springer, 2015.

[26] Jingjing Wang and Bhaskar Prabhala. Periodicity Based Next Place Prediction. In *Proceedings of the Nokia Mobile Data Challenge (MDC) Workshop in Conjunction with Pervasive*, 2012.

[27] Vincent W Zheng, Bin Cao, Yu Zheng, Xing Xie, and Qiang Yang. Collaborative Filtering Meets Mobile Recommendation: A User-centered Approach. In *Proceedings of the 24th Conference on Artificial Intelligence*, pages 236–241, Atlanta, 2010.

[28] Vincent W Zheng, Yu Zheng, Xing Xie, and Qiang Yang. Collaborative Location and Activity Recommendations with GPS History Data. In *Proceedings of the 19th International Conference on World Wide Web*, pages 1029–1038, Raleigh, 2010.

[29] Yu Zheng, Like Liu, Longhao Wang, and Xing Xie. Learning Transportation Mode from Raw GPS Data for Geographic Applications on the Web. In *Proceedings of the 17th International Conference on World Wide Web*, pages 247–256, Beijing, 2008.

[30] Yu Zheng, Xing Xie, and Wei-Ying Ma. GeoLife: A Collaborative Social Networking Service Among User, Location and Trajectory. *IEEE Database Engineering Bulletin*, 33(2):32–39, 2010.

[31] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, pages 791–800, Madrid, 2009.

[32] Changqing Zhou, Dan Frankowski, Pamela Ludford, Shashi Shekhar, and Loren Terveen. Discovering Personally Meaningful Places: An Interactive Clustering Approach. *ACM Transactions on Information Systems*, 25(3), 2007.

[33] Jun Zhou, Qinpei Zhao, and Hongyu Li. Integrating Time Stamps into Discovering the Places of Interest. In *Proceedings of the 10th International Conference on Intelligent Computing Methodologies*, pages 571–580, Taiyuan, 2014.