Privacy Leaks In Spatio-Temporal Trajectory Publishing

by

EMRE KAPLAN

Submitted to the Graduate School of Sabancı University
in partial fulfillment of the requirements for the degree of
Master of Science

Sabancı University

Spring, 2009

Privacy Leaks In Spatio-Temporal Trajectory Publishing

Approved by:

Assist. Prof. Dr. Yücel Saygın .....
(Dissertation Supervisor)

Assoc. Prof. Dr. Erkay Savaş .....
(Dissertation Co-Supervisor)

Dr. Thomas Brochmann Pedersen .....
(Dissertation Co-Supervisor)

Dr. Mehmet Ercan Nergiz .....
(Dissertation Co-Supervisor)

Assist. Prof. Dr. Cem Güneri .....

Date of Approval: .....

Privacy Leaks In Spatio-Temporal Trajectory Publishing

Emre KAPLAN

Computer Science and Engineering, Master's Thesis, 2009

Thesis Supervisor: Assist. Prof. Dr. Yücel Saygın

Keywords: Privacy, Spatio-temporal data, trajectories, data mining

Abstract

Trajectories are spatio-temporal traces of moving objects which contain valuable information to be harvested by spatio-temporal data mining techniques. Applications like city traffic planning, identification of evacuation routes, trend detection, and many more can benefit from trajectory mining. However, the trajectories of individuals often contain private and sensitive information, so anyone who possess trajectory data must take special care when disclosing this data. Removing identifiers from trajectories before the release is not effective against linkage type attacks, and rich sources of background information make it even worse. An alternative is to apply transformation techniques to map the given set of trajectories into another set where the distances are preserved. This way, the actual trajectories are not released,

but the distance information can still be used for data mining techniques such as clustering. In this thesis, we show that an unknown private trajectory can be reconstructed using the available background information together with the mutual distances released for data mining purposes. The background knowledge is in the form of known trajectories and extra information such as the speed limit. We provide analytical results which bound the number of the known trajectories needed to reconstruct private trajectories. Experiments performed on real trajectory data sets show that the number of known samples is surprisingly smaller than the actual theoretical bounds.

Mekan-Zaman Yörüngelerinin Yayınlanmasında Gizlilik Açıkları

Emre KAPLAN

Bilgisayar Bilimi ve Mühendisliği, Yüksek Lisans Tezi, 2009

Tez Danışmanı: Yar. Doç. Dr. Yücel Saygın

Anahtar Kelimeler: Gizlilik, mekan-zaman verisi, hareket yörüngeleri, veri madenciliği

Özet

Hareket yörüngeleri, hareketli objelerin içerisinde değerli bilgiler içeren zaman-mekan izleridir. Bu bilgiler çeşitli veri madenciliği uygulamalarında kullanılmak üzere toplanırlar. Şehir trafiğinin planlanması, acil ulaşım yollarının belirlenmesi ve akım takibi gibi bir çok uygulama hareket yörüngesi madenciliğinden faydalanır. Kişilere ait hareket yörüngeleri sık sık kişiye özel ve hassas bilgiler içermektedir. Bu bilgilere sahip kişiler, veriler açıklanmadan önce gerekli özeni göstermelidir. Kişisel belirteçlerin, bu veriler açıklanmadan önce temizlenmesi de bağlaç tipi saldırılara karşı zayıf kalmaktadır ve zengin arka-plan bilgileri ile bu zayıflık daha da belirginleşmektedir. Bu konuda bir alternatif, bilgileri açıklamadan önce dönüşüm teknikleri kullanarak, verilen bir küme hareket yörüngesini, ikili uzaklıkları korunacak şekilde bir

başka kümeye dönüştürmektir. Bu şekilde gerçek hareket yörüngeleri açıklanmamakta fakat ikili uzaklık bilgileri hala kümeleme gibi veri madenciliği uygulamalarında kullanılabilmektedir. Bu çalışmada, bilinmeyen ve özel bir hareket yörüngesinin, veri madenciliğinde kullanılmak üzere açıklanan ikili uzaklık bilgileri ve kullanıma müsait arka-plan bilgileri ile çözülebileceği gösterilmektedir. Bahsedilen arka-plan bilgisi, bilinen bazı hareket yörüngeleri ve hız sınırı ek bilgileri biçimindedir. Çalışmada ayrıca, özel hareket yörüngelerini çözmek için bilinmesi gereken hareket yörüngeleri sayısı hakkında analitik sonuçlar da sunulmaktadır. Gerçek hareket yörüngesi veritabanlarında yapılan deneyler bilinmesi gereken hareket yörüngeleri sayısının olması gereken teorik sınırlardan şaşırtıcı derecede küçük olduğunu göstermektedir.

To my dear mom

Acknowledgements

I wish to express my sincere gratitude and thank to Dr. Yücel Saygın, for his continuous support, guidance and help in both my thesis and graduate study as well as our conference and journal paper, without his support, this study could not be done. He has been helpful and understanding to me and care for my studies. I also thank to him because of sending me to conferences from which i have benefited in terms of academic development.

I would like to thank Dr. Erkay Savaş, Dr. Thomas B. Pedersen and Dr. M. Ercan Nergiz for their great support, guidance, patience and great ideas they brought into our conference and journal paper and also to my thesis. I especially thank Thomas B. Pedersen for his contribution in writing this thesis.

I appreciate Ismail Fatih Yıldırım for his help in LyX.

I would like to thank the thesis committee for their comments.

Last, but not the least, to my family, especially to my dear mother because she was the best supporter of me throughout the graduate studies as she always does.

Contents

List of Figures

1 Introduction

Today's world highly benefits from the wireless technologies which are in demand for the last 15 years. Since mid 90's, wireless technologies have been developed together with various services and applications. As a result, people became more and more mobilized with mobile phones and the wireless trend is widely adopted. Today, mobile phones and personal digital assistants (PDA) are also equipped with Global Positioning Systems (GPS), Wi-fi, Bluetooth, and Radio Frequency Identification (RFID) tags. As these technologies become part of our life, a lot of time-referenced location information is collected by different mobile service providers. Time-referenced location data is also called spatio-temporal data, where "spatio" means the location, and "temporal" means related to time. The collected spatio-temporal data contains valuable information to be harvested by spatio-temporal data mining techniques for various applications such as traffic management, identification of evacuation routes, trend detection, geo-marketing, and sometimes for geo-spamming. In general, spatio-temporal data can be analyzed to discover new patterns about how people travel and behave in public areas. However, privacy is a major concern for data mining in general and spatio-temporal data mining in specific. Privacy issues become even more serious when the data miner and the data owner are different, and they have to share the data for data mining, or when organizations publish their data for public use. For example, GSM companies with the capability of collecting huge amounts of spatio-temporal data about their users which shows where they have been at specific times and what are their habits etc. Therefore, companies who col-

lect privacy sensitive spatio-temporal data also have the obligation to protect the privacy of their customers. Therefore, they need mechanisms to harvest useful information from their data while protecting the privacy of individuals. Privacy concerns are not solely the subject of spatio-temporal datasets but also in data mining context for conventional data. However, existing methods, developed for conventional data such as tabular data cannot be directly applied to spatio-temporal datasets. Moreover, privacy threats for spatio-temporal data are different and more diverse than conventional data. In order to preserve individual privacy, the first attempt would be to remove the personal identifiers from the spatio-temporal data collected before it can be disclosed to third parties or made public. However, this has been shown that simply removing the identifiers is not enough, since some of the remaining attributes in the data set could be used to join the public and private information as discussed in [48]. Availability of rich background information makes the privacy issues more complicated [50] and this is even more serious in case of spatio-temporal data considering the variety of background information to be used. A safer approach may be to perturb the data before it is made public, as discussed in [40], or apply transformations on the data in a way that preserves important properties such as pair wise distances among the data points so that data mining techniques such as clustering can still be applied. However, there may still be privacy risks in such transformations on spatio-temporal data. In this thesis, we consider distance preserving data transformations on trajectories and identify the privacy leaks in such transformation when an adversary has background information. We propose two different methods to show that private trajectories could be discovered with

very high precision. For both methods, we consider the following scenario where the adversary would like to reconstruct the movement behavior of a specific individual (the so called "target trajectory") and knows the released set of pair wise distances between the trajectories in the dataset. We assume that the trajectories themselves are not published but only their pair wise distances are available for data mining purposes. In both methods, we show that target trajectories can be found very precisely with limited background information and using computationally feasible methods. Contributions of this thesis are listed below:

1. We have shown that publishing pair wise distances between data points leads to privacy leaks.

2. Easy accessible background information in addition to pair wise distances enhances the attack quality and results in very precise solutions leading to further privacy leaks.

In the first method, the adversary has access to some background information such as the average speed, or the maximum speed of the trajectory and a bunch of trajectories from the dataset together with access to released pair wise distances. The adversary reconstructs the target trajectory with a heuristic based algorithm. Due to limited information, he/she may not exactly recover the target trajectory but can still figure out the basic shape of the route. Experiments show that even with very limited information in terms of number of trajectories, the adversary can reconstruct the target trajectory precisely in most of the cases. As a qualitative measure we discuss

the relation between reconstruction quality of the target trajectory and the number of trajectories the adversary has.

In the second method, the adversary only has a bunch of trajectories and access to published pair wise distances between trajectories of the dataset which is going to be attacked. This method is not solely an alternative attack method but has advantages over the first one in the sense that:

1. It is faster than the previous method.
2. Target trajectory can be discovered and the solutions are mathematically sound however, due to the limited information, one can generate various numbers of candidates which are all identical in terms of distances to the known trajectories and equally possible to be the target trajectory.
3. This method results in a set of trajectories which enables us to do several other observations such as the confidence about the areas that the target trajectory possibly passed through. Area where the most of the results are passing through is likely to be also passed by the target trajectory. We calculate confidence of those areas in terms of number of results found in the attack.

The thesis is organized as follows: Section 2 focuses on background information and related work done in the area of privacy and spatio-temporal data mining. In Section 3, we cover the definitions, notations and the formal problem definition addressed in this thesis. In Section 4 we discuss our

first attack method, in Section 5 we discuss our second attack method. We conclude in Section 7 and discuss the future works.

2 Background and Related Work

The contributions of this thesis are in the research area of privacy in spatio-temporal data mining. In this chapter, we survey several other related research areas as well. These are privacy preserving data mining, anonymization and privacy in spatio-temporal data. In this section, we provide the basics and related work about these topics. We start by defining what is privacy and why it is important for this research and continue covering other topics listed above.

2.1 Privacy In Data Mining

Privacy is defined in [4] as,

> "Privacy is the ability of an individual or group to seclude themselves or information about themselves and thereby reveal themselves selectively."

Privacy rights are developed almost in all countries so that utilizing personal data is not an easy task. Disclosing information other than what the data owner has selected to share is called *privacy leak* and should be avoided. Privacy is important because of the growing number of emerging technologies in mobile area, vast number of spatio-temporal data is collected and data mining applications harvest valuable information out of those huge data heaps. The key point is: does the collected information contain more than what

the individual decide to disclose? For example, no one can be traced unless stated otherwise but what about collected GPS data or RFID tag data of that individual? Are those data points just innocent points or more than that? In spatio-temporal data mining, collected data points are utilized in various techniques to decide or create meaningful results out of those considerable amounts of data. When information from various sources come together, unexpected data disclosure may result and cause privacy leaks. In [18, 19, 34] authors discuss when the results of data mining applications cause privacy violations.

2.2 Privacy Preserving Data Mining

Privacy preserving data mining is a very popular research area since the beginning of 2000's. A large number of collected data needs well designed algorithms to be utilized. As discussed in 2.1, utilizing personal information is not an easy task. Privacy preserving algorithms are needed to preserve individual privacy while utilizing data as in ordinary data mining techniques. Agrawal and Srikant's work [9] in this area lead other researchers to focus on privacy aspects of data mining. The basic method in privacy preserving data mining is to remove personal identifiers and other sensitive information in a way that the data will not be linked to its owner. In [49], authors worked in association rule hiding such that sensitive information is concealed. Another way is to do Secure Multi-party Computation (SMC) on vertically or horizontally partitioned data. In [32, 30], authors developed such methods based on SMC.

2.3 Privacy In Spatio-Temporal Data

Privacy in spatio-temporal data is another popular research area in recent years. Location based services (LBS) are very popular in today's world which are discussed in [10, 14, 41, 15]. Mobile applications often use location information and service according to client's location. In [10], authors propose a way to express users preferences on their location information and obfuscate the data to preserve privacy.

Data perturbation is a well-studied field in data privacy in[10, 33, 37, 40] which researchers studied on perturbing the data in a way that preserve privacy. Indeed, several other papers have treated this topic, showing that data perturbation techniques are not always effective in protecting privacy [37] because the perturbation can be predictable.

Anonymization is another research subject aiming at protecting individual privacy. Anonymization based privacy preserving methods are discussed in [48, 8, 46, 6, 23, 14]. Techniques for trajectory anonymization were recently proposed in [8] and [46], but privacy risks after data release were not considered. In another recent work, privacy risks due to distance preserving data transformations were identified [51], however spatio-temporal data was not addressed. The privacy risks in trajectory data was addressed in [50] where authors point out how parts of a trajectory could be used as quasi-identifiers to discover the rest of the trajectory. In this work, authors assume that the trajectories are distributed vertically across multiple sites where sites are curious to learn the rest of the trajectory, and the authors propose

methods to prevent that by suppressing parts of the trajectories before they are published.

3 Preliminaries and Problem Definition

In this section, we provide the basic building blocks for the thesis. The methods covered in the following sections 4 and 5 uses the notations and definitions expressed in this section. We first define the sample points, trajectory and the notations, then we discuss the distance metrics and dissimilarity matrix. Background information used in both methods are discussed followed by the discussion of error and success rate measures. At last, we discuss the gradient descent algorithm which is adopted our method discussed in Section 4.

3.1 Trajectories and Their Properties

Trajectories are paths in space-time. For every time frame, there is a space (or say location) coordinates which are easily collected by GPS devices or mobile phones nowadays. Hence, a trajectory can be represented as a set of points, denoted as *sample points*. A sample point has 3 features namely, x-coordinate, y-coordinate and time stamp, t. In this thesis, a discrete *trajectory* is defined as a polyline represented as a list of sample points which the time component is discarded for the sake of simplicity. We represent trajectories in two different forms. A trajectory either represented as a column vector which is described below or as a list of sample points. If we have more than one trajectory then i^{th} trajectory is denoted as T^i. The list of sample points representation of a trajectory is as follows:

$$T = ((x_0, y_0, t_0), ..., (x_{n-1}, y_{n-1}, t_{n-1}))$$

In this thesis, we assume that trajectories

1. are aligned.

2. have constant sampling rate ($t_{i+1} - t_i = \triangle t$, for some constant $\triangle t$).

Algorithms for ensuring these conditions can be found in [28]. Time component is discarded for the sake of simplicity and there are methods to have constant sampling rate which makes time component redundant, so we represent a trajectory as a list of (x, y) coordinates (or a vector in R^{2n}). We write T_i to represent the i^{th} sample-point (x_i, y_i, t_i). In most of this thesis, we think of a trajectory as a column-vector in a large vector-space. We use calligraphic letters to refer to the vector representation of a trajectory. The vector representation of a trajectory T is: $\mathscr{T} = (x_0, y_0, t_0, \ldots, x_{n-1}, y_{n-1}, t_{n-1})^T \in \mathbb{R}^{3n}$. In this case $\mathscr{T}_i$ is the i^{th} element of the vector (i.e. $\mathscr{T}_0 = x_0, \mathscr{T}_1 = y_0, \ldots, \mathscr{T}_{3n-1} = t_{n-1}$). A trajectory $\mathscr{T}$ can posses many properties which are of interest in different situations, such as maximum and average speed of a trajectory, closest distance to certain locations, duration of longest "stop", or percentage of time that $\mathscr{T}$ moves "on road". We use average and maximum speed property of trajectories in the first method in Section 4, so we discuss these properties in the following subsections..

In this thesis, we also define known, target, candidate and reconstructed trajectory. Definitions of those trajectories are below.

Known trajectory is the trajectory known by the adversary to be used in the privacy attacks as the key information. In our methods, the adversary generally has a bunch of trajectories known from the dataset where s/he attacks to.

Target trajectory (or private trajectory) is the trajectory which the adversary wants to find. Target trajectory is naturally a member of the dataset where the adversary attacks to. Target trajectory denoted as T' and $\mathscr{X}$.

Reconstructed trajectory is the trajectory which is the only outcome of the attack. This term is used in the first method that we discuss in Section 4.

Candidate trajectory is the trajectory which is one of the outcomes of the attack. Due to limited information, there are more than one trajectory in general, satisfies the conditions in terms of knowledge of the adversary. Candidate trajectories are possible representatives of the target trajectory although they may not be similar on a map but mathematically they satisfies all the conditions as the target trajectory does and none of them is better than the others. Candidate trajectory i is denoted as c_i. This term is used in the second method that we discuss in Section 5.

3.2 Distance Metrics and Dissimilarity Matrix

3.2.1 Distance Metrics

The most important property of the trajectories that our work relies on, is the distance property. *Distance* is the numeric value which defines the closeness of two data points. The distance metric can be any of the metrics defined below but for the sake of simplicity we chose Euclidean distance. However when comparing trajectories in terms of their relative distances, using real Euclidean distances or distance square values doesn't matter since we care of the relative distances of those points hence we use Euclidean square distance defined below. Moreover omitting square root operation saves computation results in speed because methods presented in this thesis extensively do this operations. There are some distances metrics in the literature[45]. The distance metrics used throughout this work is listed below. Note that Euclidean distance and P-norm distance are using the vector representation so that their indices are in range of $[0,\ 2n-1]$ while rest of the distance metrics are using list of sample points representation so that their indices are in the rage of $[0,\ n-1]$

Euclidean distance

$$\|\mathscr{T} - \mathscr{T}'\|_2 = \sqrt{\sum_{i=0}^{2n-1} |\mathscr{T}_i - \mathscr{T}_i'|^2} \tag{3.1}$$

P-norm distance

$$\|\mathscr{T} - \mathscr{T}'\|_p = \left(\sum_{i=0}^{2n-1} |\mathscr{T}_i - \mathscr{T}_i'|^p \right)^{1/p} \tag{3.2}$$

Average p-norm distance

$$ASD_p(T, T') = \frac{1}{n} \sum_{i=0}^{n-1} \|T_i - T_i'\|_p \tag{3.3}$$

Variance distance

$$d_v(T, T') = \frac{1}{n} \sum_{i=0}^{n-1} (\|T_i - T_i'\|_2 - d_2(T, T'))^2 \tag{3.4}$$

Euclidean-Square distance

$$\|T - T'\|^2 = \sum_{i=0}^{n-1} |T_i - T_i'|^2 \tag{3.5}$$

Area distance $d_A(T, T')$, which is the area of the region enclosed between the two trajectories [45].

Note that the average sample distance is a special form of average p-norm distance where $p = 2$. In this thesis, we mainly focus on Euclidean distance and square of Euclidean distance which we defined above.

3.2.2 Dissimilarity Matrix

Data holders want to do privacy preserving data mining so that instead of publishing actual data points (trajectories in this thesis) they publish the

pairwise distances of the data points so the information in terms of data mining is contained in the distances in-between data points. The list of pairwise distances of all trajectories to each other are represented in the matrix form, where $D(i, j)$ represents the distance between i^{th} and j^{th} trajectory in the matrix D. This is called *dissimilarity matrix* also mentioned in [35, 36, 51]. An example of a dissimilarity matrix is given below.

Dataset:	Distances:
Trajectory 1: [(1,1)(2,2)(3,3)]	D(2,1) = $\sqrt{3}$
Trajectory 2: [(2,1)(3,2)(4,3)]	D(3,1) = $\sqrt{15}$
Trajectory 3: [(2,3)(3,4)(4,5)]	D(2,3) = $\sqrt{12}$

Dissimilarity Matrix, D

1	2	3
0	$\sqrt{3}$	$\sqrt{15}$
-	0	$\sqrt{12}$
-	-	0

Figure 1: Dissimilarity Matrix

Dissimilarity matrix is denoted as D and a pairwise distance between trajectory i and trajectory j is denoted as $D(i, j)$.

3.3 Types Of Background Information

3.3.1 Known Distances

The most important property of the trajectories that we use in this thesis is the known distances. Known distances are in the form of dissimilarity matrix defined in Section 3.2.2 which is the only released information that includes

the distance from an unknown trajectory $\mathscr{T}$ to a fixed trajectory, $\mathscr{T}'$ for all trajectories in the dataset and accessible by the adversary. This data may be release in order for a third party to perform clustering on the trajectories. Calculating distance in between trajectories is discussed in Section 3.2. When using a continuously differentiable norm to compute the distance between $\mathscr{T}$ and $\mathscr{T}'$ we obtain a continuously differentiable property of $\mathscr{T}$; e.g. $\Delta_{\mathscr{T}'}(\mathscr{T}) = d(\mathscr{T}', \mathscr{T})$ is continuously differentiable. Note that all these distance measures are continuously differentiable with the exception of p-norm distance for odd p.

3.3.2 Average and Maximum Speed

Another property of trajectories, which is natural to consider, is the maximum or average speed at which the moving object is traveling. Since we only have discretized versions of the trajectories, with sample points taken at a fixed sample rate, we can only approximate the average and maximum speed:

$$\text{avgSpeed}(T) = \frac{1}{n-1} \sum_{i=0}^{n-1} \frac{\|T_i - T_{i+1}\|_2}{\Delta t} \tag{3.6}$$

$$\text{maxSpeed}(T) = \max_i \left\{ \frac{\|T_i - T_{i+1}\|_2}{\Delta t} \right\} \tag{3.7}$$

where Δt is the known, constant sample rate (which we have discarded from the description of the trajectory itself). Note that the average/max speed in this case is approximated by the average/max speed of each *segment* of the discretized trajectory, where segment i is the line segment (T_i, T_{i+1}),

or, when written in the vector notation: $((\mathscr{T}_{2i}, \mathscr{T}_{2i+1}), (\mathscr{T}_{2i+2}, \mathscr{T}_{2i+3}))$. The average speed is easily seen to be continuously differentiable. To compute the derivative of the maxSpeed, first note that the derivative of the maximum function can be approximated as:

$$\frac{\partial}{\partial x_i} \max\{x_0, \ldots, x_{n-1}\} = \begin{cases} 1 & \text{for } i \in \text{argmax}_i\{x_0, \ldots, x_{n-1}\} \\ 0 & \text{else} \end{cases} \tag{3.8}$$

where $\text{argmax}_i\{x_0, \ldots, x_{n-1}\} = \{i_1 \ldots, i_l\}$ is the set of indices such that x_{i_j} has the largest value of $\{x_0, \ldots, x_{n-1}\}$ (more than one element can have the maximum value).

When there is more than one largest argument to the max function, the partial derivatives with respect to those arguments are not well-defined (the right-derivatives are 1, while the left-derivatives are 0). However, in the following, we will use the convention that the partial derivatives of the largest arguments are 1 in those arguments.

Let S be the set of indices of the first sample points on the fastest segments of the trajectory: $S = \text{argmax}_i\{\|T_i - T_{i+1}\|_2/\Delta t\}$, and let $\mathcal{S}_t = \{2s + t | s \in S\}, t \in \{0, \ldots, 3\}$ be the sets of the indices of the coordinates of the vector representation of the fastest segments ($\mathcal{S}_0$ is the set of x-coordinates on the first sample points, $\mathcal{S}_1$ is the set of y-coordinates on the first sample points, $\mathcal{S}_2$ is the set of x-coordinates on the second sample points, etc.). In the following we will use a generalization of Kronecker delta: $\delta_{i,\mathcal{S}}$, which is 1 if

$i \in \mathcal{S}$, and 0 otherwise. The partial derivatives of the maximum speed is:

$$\begin{aligned}
\frac{\partial}{\partial \mathscr{T}_i}\text{maxSpeed}(T) &= \frac{\partial}{\partial \mathscr{T}_i} \max_j \left\{ \frac{\|T_j - T_{j+1}\|_2}{\Delta t} \right\} \\
&= \sum_{k=0}^{3} \delta_{i,\mathcal{S}_k} \frac{1}{\Delta t} \frac{\partial}{\partial \mathscr{T}_i} \|(\mathscr{T}_{i-k}, \mathscr{T}_{i-k+1}) - (\mathscr{T}_{i-k+2}, \mathscr{T}_{i-k+3})\|_2 \\
&= \sum_{k=0}^{3} \delta_{i,\mathcal{S}_k} \frac{1}{\Delta t} \frac{\mathscr{T}_i - (-1)^{\delta_{k,\{0,1\}}} \mathscr{T}_{i+2} - (-1)^{\delta_{k,\{2,3\}}} \mathscr{T}_{i-2}}{2\|(\mathscr{T}_{i-k}, \mathscr{T}_{i-k+1}) - (\mathscr{T}_{i-k+2}, \mathscr{T}_{i-k+3})\|_2}
\end{aligned}$$

This partial derivative is not continuous, however, as we argue in Section 4.4, it is still suitable for the reconstruction of trajectories.

3.4 Error and Success Rate of a Trajectory

3.4.1 Error

We define the "error" of a candidate $\mathscr{X}'$ as

$$E(\mathscr{X}') = \frac{1}{2} \sum_{i=1}^{m} \left(P_i(\mathscr{X}') - P_i(\mathscr{X}) \right)^2 \tag{3.9}$$

where P_i are the properties which are known about the target trajectory (in other words: $P_i(\mathscr{X})$ are known values). The error function is the difference between the given properties of the target trajectory, $\mathscr{X}$, and corresponding properties of the candidate trajectory, $\mathscr{X}'$. The error function is 0 when the properties of the candidate and the target trajectory are the same and a positive number, otherwise. Furthermore, the error function is positive. It is differentiable as long as the properties are differentiable.

The error function is defined for the adversary to measure how well s/he uses background information. It depends on only the information that the adversary has.

When the adversary uses the hyper-lateration technique with an adequate number of trajectories, it exactly finds the target trajectory. But in general, due to lack of information, the target trajectory may not be found exactly so the adversary needs to calculate the difference between the target trajectory and the candidate trajectory in terms of their properties. This measure is called *error* and error of a reconstructed trajectory shows how much the reconstructed trajectory differs from the target trajectory. Any zero of the error function exhausts the knowledge (i.e the known properties) about the target trajectory. Equation 3.9 is a general error definition defined over trajectories with any number of properties. In Equation 3.10, error of a candidate trajectory is calculated over its only known property, the distances.

$$E(X') = \sum_{i=1}^{i=n} \left(||X' - T_i||_2 - \delta_i\right)^2 \; where\, \delta_i = ||X - T_i||_2 \tag{3.10}$$

In Equation 3.10, error is defined over the known distances. It shows how well the candidate trajectory, X', satisfies known distances to all the T_i's. It calculates the squared differences between the distance from candidate trajectory, X', to known trajectory, T_i, and δ_i. Note that δ_i represents the distance of the known trajectory T_i from the target trajectory X. This tells us something about how far the candidate trajectory is from the target

trajectory. The error function in 3.10 is 0 when the candidate trajectory is at δ_i to the known trajectory T_i for all $i \in \{1, .., n\}$, and a positive number otherwise. Figure 2 depicts how to find error on a $2d$ plane on ordinary points. The logic behind the calculations is the same in trajectories because trajectories with n sample points are just points in R^{2n}. In Figure 2, black point denotes the candidate trajectory, gray point denotes the known trajectory and finally red point denotes the target trajectory. According to Equation 3.10, summation calculates $d1$ and δ_i calculates $d2$ so the overall error equation, $E(X')$, calculates $d1 - d2$ which denotes the error between the target trajectory and the candidate trajectory. This is shown in Figure 2.

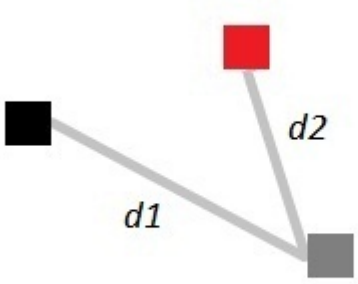


Figure 2: Error

3.4.2 Success Rate

In essence the success depends on how well the candidate represents the target.

In [35] an unknown target trajectory was reconstructed from knowledge of the distance from the target trajectory to each trajectory in a set of known trajectories. To evaluate the success of the reconstruction the following success rate was used:

$$SR(\mathscr{X}') = 1 - \frac{\|\mathscr{X} - \mathscr{X}'\|_2}{\delta_{min}} \tag{3.11}$$

where $\delta_{min} = \min_i(\delta_i)$ is the smallest known distance. This success-rate is 1 if the method finds $\mathscr{X}$ precisely, 0 if it returns the closest known trajectory, and negative if it performs worse than just returning the closest known trajectory. This measure has a number of shortcomings, which makes it difficult to compare the success of different algorithms, or even the same algorithm, but applied to different datasets. One obvious problem is that the success rate cannot be applied to reconstruction methods which do not use the distance to known trajectories. Furthermore, it is very difficult to obtain a high success rate for a dataset with many close trajectories (since δ_{min} is likely to be a very small number). Another problem is that this success does not take the "resolution" of the target trajectory into account: For fixed length target trajectories the success rate does not depend on the number of sample points. If the target trajectory has a high sample rate (high resolution) it is likely that the quality of the reconstruction is more sensitive to noise than if the same target trajectory only has a low resolution.

We overcome some of the shortcomings of the old success measure by defining a new *success rate* $SR(\mathscr{X}')$ of a candidate trajectory. The *success rate* should satisfy the following properties:

- $SR(\mathscr{X}') \in [0, 1]$

- $SR(\mathscr{X}) = 1$
- Depend only on the target and candidate trajectories.
- Be independent of the magnitude of coordinates.

Intuitively the quality of a candidate trajectory depends on how far away the candidate trajectory is from the target trajectory at any given time. In our case, since we assume that trajectories are aligned, the average distance of the candidate trajectory to the target trajectory over time is the average sample distance:

$$ASD_T(X) = \frac{1}{n}\sum_{i=0}^{n-1} \|X_i - T_i\|_2 \tag{3.12}$$

The average sample distance alone, however, is not a good measure of success, since it depends highly on the magnitude of the coordinates. To factor out this dependency on the magnitude of the coordinates, we divide the average sample distance with the total length of the target trajectory, which can be computed as:

$$\|T\|_l = \sum_{i=0}^{n-2} \|T_i - T_{i+1}\|_2 \tag{3.13}$$

The fraction $ASD_T(X)/\|T\|_l$ is a non-negative real number, which is 0 when $X = T$. We define the success rate from this fraction as follows:

$$SR(X') = \mathrm{e}^{-\alpha ASD_T(X)/\|T\|_l} \tag{3.14}$$

where α is a sensitivity factor which decides how steep the success rate goes to 1 as the candidate approaches the target. The new success rate satisfies the

criteria listed above: $SR(T) = \mathrm{e}^{-\alpha ASD_T(T)/\|T\|_l} = \mathrm{e}^0 = 1$, and as $ASD_T(X)$ tends to infinity, $SR(X)$ tends to $\mathrm{e}^{-\infty} = 0$.

The success measure defined above may not be appropriate in all situations for instance when the trajectories may be laying on top of each other, thus giving the visual impression of a perfect match, but may be very far apart in time: Even though all sample points overlap, the chronological ordering may be reversed, this situation will give a very poor success rate with the measure defined above, but will appear as a perfect match.

All in all, success rates aim to evaluate the attack methodology in the point of view of the researcher. That is, both success rates involves the target trajectory which is an unknown information to the adversary. Success Rate 1 depends on the target trajectory, candidate trajectory and some known distances from the dissimilarity matrix (i.e the the distance of the closest known trajectory to the target). Success Rate 2, however, depends only on the target and the candidate trajectory and overcomes the shortcomings discussed above.

Lower Bound of Success Rate Although the adversary cannot know the final success rate before the attack, there are situations where s/he can give a lower bound on the success rate. Since the success rate is defined in terms of the average sample distance, s/he can get the following bound in the situation where s/he knows the average sample distance to a set of known trajectories.

Theorem 1 *Let $T^1, \ldots, T^m$ be known trajectories, and let $\delta_i = d_2(T^i, X)$ be the average sample distances to the unknown trajectory X. Then, for any trajectory X' with $E(X') = 0$ the success rate is:*

$$SR(X') \geq \mathrm{e}^{-2\alpha\delta_{max}/(n\|X\|)} \tag{3.15}$$

where $\delta_{max} = \max_i(\delta_i)$ is the largest given distance, and E is the error function defined in Eq. 3.10.

While the attacker does not know the length of X, he may be able to estimate it from his background knowledge.

Proof . We first observe that since E is a sum of the non-negative terms $1/2(d_2(T^i, X') - \delta_i)^2$, and since $E(X') = 0$, necessarily $d_2(T^i, X') = \delta_i$.

Now, note that for all $k \in \{1, \ldots, m\}$

$$\begin{aligned}
ASD_X(X') &= \frac{1}{n}\sum_{i=0}^{n-1} \|X'_i - X_i\|_2 \\
&= \frac{1}{n}\sum_{i=0}^{n-1} \|X'_i - T_i^k + T_i^k - X_i\|_2 \\
&\leq \frac{1}{n}\sum_{i=0}^{n-1} (\|X'_i - T_i^k\|_2 + \|T_i^k - X_i\|_2) \\
&= \frac{1}{n}(d_2(T^k, X') + d_2(T^k, X)) \\
&= \frac{2\delta_k}{n}.
\end{aligned}$$

Inserting this in the definition of the success rate gives us:

$$SR(X') = \mathrm{e}^{-\alpha ASD_X(X')/\|X\|_l} \geq \mathrm{e}^{-2\alpha\delta_k/(n\|X\|_l)} \tag{3.16}$$

Since Eq. 3.16 is true for all δ_k, it is true for δ_{max}. □

3.5 Problem Definition

Location information is collected by GSM companies and the adversary also has a mobile phone and has his/her own trajectories in the dataset. This way s/he naturally has some information from the dataset. S/he also knows the pairwise distances of his/her trajectories to the other trajectories because that information is contained in the dissimilarity matrix defined in Section 3.2.2 which is published for data mining purposes hence s/he will be capable of calculating target trajectories from the dataset.

Formally, given a dataset D, with cardinality c, and a set of trajectories from this dataset, say k trajectories (and $k \leq m$), and pairwise distances of those k trajectories to the target trajectory, then the adversary can compute the target trajectory. In [51], a target trajectory can be found by hyperlateration technique (see Section 5.3.2) but it requires a high number of known trajectories, which may be infeasible. Assume that trajectories in the data set have 500 sample points, the method presented in [51] needs $2n + 1$ trajectories where n is the number of sample points in trajectories, hence $2 * 500 + 1 = 1001$ trajectories and corresponding pairwise distances

are needed. However knowing high number of trajectories and their pairwise distances may be infeasible. The first method discussed in Section 4 aims to lower this high number of known trajectories needed to be known by the adversary. Actually $2n + 1$ known trajectories is indeed not needed in many cases because a heuristic based method can find very precise candidate trajectory with a very limited information.

We built another method discussed in Section 5 on top of the hyperlateration technique where we use interpolation technique discussed in Section 5.3.1 in between sample points of a trajectory so that we lower the need of high number of trajectories because of the facts that the trajectories are aligned and constant sampled as discussed in Section 3.1. Trajectory data is mostly collected by GPS data, which are from mobile vehicles, that follows up certain paths and roads. When the resolution of the data is low, trajectories have high linear dependencies between the sample points that follows each other. In this fashion, if we know two points, we can guess the other points in-between those points because the sampling is constant (i.e time difference between points is constant) and it is possible to guess more or less how long distance covered during the time between two sample points hence the points in-between can be predictable up to some degree depending on the structure of the trajectory. As the trajectory is more linear, the trajectory or part of the trajectory can be more predictable. We use linear interpolation for the sake of simplicity and trajectories themselves are set of lines because they are denoted as a set of sample points so each pair of sample points forms a line so that the points in-between those sample points should be on that line unless the collected data is noisy.

4 Discovering Private Trajectories Using Background Information

4.1 Introduction

In this attack method, we consider the following scenario: A malicious person wishes to reconstruct the movements (the "target trajectory") of a specific individual. Besides a released set of mutual distances between a data set of trajectories, which contains the target trajectory, the attacker has some background information, such as the average speed or maximum speed of the trajectory, and some of the other trajectories in the data set. We propose a concrete algorithm which can reconstruct the target trajectory from this information.

We demonstrate that trajectories can be reconstructed very precisely with very limited information using relatively simple methods. In particular we apply our method to two real-world data-sets. In one data-set, containing the trajectories of private cars in Milan, we can reconstruct an unknown trajectory with 500 sample points by knowing its distance to only 60 known trajectories. This is in sharp contrast to the 1001 known distances which would be needed to solve the corresponding system of equations to find the unknown trajectory.

We propose a method which can reconstruct trajectories from a very wide range of continuous properties (see Section 2); the method of known

distances is only a special case. We show that any property of $\mathscr{T}$ which can be expressed as a continuously differentiable function $f : \mathbb{R}^{2n} \to \mathbb{R}$ can be used to reconstruct $\mathscr{T}$. Our method is optimal in the sense that it will eventually find a candidate which exhausts all the information available about the target trajectory.

4.2 Reconstructing Trajectories

We consider how a malicious person can find an unknown trajectory, X, with as little information as possible. Any information we have about X may improve our ability to reconstruct X; a car does not drive in the ocean, and rarely travels at a speed of more than 200 km/h. The information which the malicious person has about a trajectory can be divided into two kinds:

1. Data which has been released into the public domain by a data holder (in some anonymized format)
2. Background information which the malicious person already had about the trajectory.

In this thesis, the only kind of released information we address are the mutual distances between trajectories. This data may be release in order for a third party to perform clustering on the trajectories. Examples of background information are maximum speed of a trajectory, since speed limits are well-known.

With a sufficient number of known properties of X, the trajectory can be fully reconstructed. For example, if $2n$ linear properties of X are known, we have a system of $2n$ linear equations. Solving these $2n$ equations gives us the exact unknown trajectory. The number of linear properties we need to know, however, is at least as large as the number of coordinates in the trajectory itself. If only $m \ll 2n+1$ linear properties are known, the solution will be in a $(2n-m)$-dimensional subspace, at best. When the candidate can only be restricted to a subspace, it can be arbitrarily far away from X. If the known properties are non-linear, finding a solution to the corresponding equations, even if sufficient number of properties is known, may even become infeasible.

As an example, consider m known trajectories, $\mathscr{T}^1, \ldots, \mathscr{T}^m$, and m corresponding positive real values δ_i, where

$$\delta_i = \|\mathscr{X} - \mathscr{T}^i\|_2 \tag{4.1}$$

for unknown trajectory $\mathscr{X}$. Our task is to find an approximation $\mathscr{X}'$ which minimizes the distance $\|\mathscr{X} - \mathscr{X}'\|_2$. This can be done by hyper-lateration, a generalization of trilateration. By squaring the known distances we obtain a system of n quadratic equations: $\delta_i^2 = \sum_{i=0}^{2n-1} |\mathscr{T}_i - \mathscr{T}_i'|^2$, for $i \in \{1, \ldots, n\}$. However, by subtracting each of these equations from the first equation we obtain $n-1$ linear equations:

$$\begin{aligned} \delta_1^2 - \delta_i^2 &= \|\mathscr{X} - \mathscr{T}^1\|_2^2 - \|\mathscr{X} - \mathscr{T}^i\|_2^2 && (4.2) \\ \Rightarrow \delta_1^2 - \delta_i^2 &= \sum_{j=1}^{2n} 2\mathscr{X}_j(\mathscr{T}_j^i - \mathscr{T}_j^1) + (\mathscr{T}_j^1)^2 - (\mathscr{T}_j^i)^2 && (4.3) \end{aligned}$$

for $i \in \{2, \ldots, 2n+1\}$. As previously argued, this approach is unsatisfactory since we need to know at least $(2n+1)$ distances where a trajectory may have thousands of sample points so that obtaining $(2n+1)$ distances is infeasible in many cases, and the method is too sensitive to noise.

As seen from the discussion above, we need a method which can reconstruct the unknown trajectory with considerably fewer known properties than coordinates. However, the best thing is to find a candidate trajectory which has the same properties as the properties we know about $\mathscr{X}$. If, for instance, the only information we have about $\mathscr{X}$, is a car driving at an average speed of 50 km/h in Athens, then any $\mathscr{X}'$ which moves along the roads of Athens at 50 km/h is a possible solution. We use the error measure defined in Section 3.4 to calculate the difference between the given properties of $\mathscr{X}$, and the corresponding properties of the candidate $\mathscr{X}'$; in the case above, the distances to the known trajectories because we want to minimize the error so that the candidate trajectory with lowest error is simply the best candidate trajectory.

4.3 Noise

The information available to the attacker about the unknown trajectory may not always be precise but subject to noise and the noise is either a deliberate attempt from the data holder to anonymize the released data, or simply errors in the background knowledge of the attacker such as dissimilarity matrix, known trajectories or average / maximum speed.

It is important to note the difference between noise in the original measurement of trajectories, and noise which is added before data is released. Noise due to measurements does not affect the attack method but affects the quality of the data because for instance if there is a noise in trajectories, then the dissimilarity matrix built also includes that noise and the distance values will be according to the noisy data so that the the trajectory we reconstruct will not be the real target trajectory without noise but the target trajectory in the dataset which is noisy. Noise can also be added to published data deliberately to prevent beaching the privacy of individuals. Data perturbation is a well-studied field in data privacy [40]. As an example of deliberately added noise, consider a trajectory database which releases the pairwise distances of all trajectories that it contains. The distances of these trajectories have to satisfy the triangle inequality however, if the noise is added independently to each of the released distance, the distances will no longer satisfy the triangle inequality.

We consider the case of known distances, where the attacker knows m trajectories, $\mathscr{T}^1, \ldots, \mathscr{T}^m$, and m corresponding distances:

$$\delta_i = \|\mathscr{X} - \mathscr{T}^i\|_2 + \epsilon_i \tag{4.4}$$

where ϵ_i are *noise terms*.

When the equations known to the attacker have errors as above, reconstruction based on solving the system of equations by hyper-lateration as described in Section 4.2 does not work well. On the other hand, if the noise follows a distribution with an expected value of 0, a reconstruction method

based on optimization should still perform well, since the real solution is likely to be close to the solution of the erroneous equations. In Section 4.5, we show that our method can handle additive noise which follows a Gaussian distribution up to a certain standard deviation.

4.4 Our Reconstruction Method

We adopt the steepest descent (gradient descent search) algorithm to find a candidate with minimum error.

The error-function in Equation 3.10 has value 0 exactly when the candidate trajectory has the same properties as the known trajectory $\mathscr{T}_i$, for all properties $P_i, i \in \{1, \ldots, m\}$. Furthermore, since Equation 3.10 is a positive valued function, the target trajectory is a global minimum. There may, however, be more than one global minimum, as well as several local minima; but any zero of the error-function exhausts the knowledge we can possibly have about the unknown trajectory, given the known properties. Recall that the gradient descent algorithm finds a zero of a positive and continuously differentiable function E as follows

1. Choose a random point, x_0, in the domain of E.
2. Iteratively define $x_{i+1} = x_i - \gamma \nabla E(x_i)$, for some step-size $\gamma > 0$.
3. When $x_{i+1} = x_i$ ($\nabla E(x_i) = 0$) a (local) minimum has been reached. If $E(x_i) = 0$ we have a global minimum (since E is non-negative), and we stop. Otherwise, we go back to step 2.

Note that the size of the steps taken in the direction of the gradients are determined by the step size, γ. Ideally, the attack should neither underestimate nor overestimate the step size. If the step size is too small, the attack will converge very slowly, thus yielding poor success rate, whereas if the step size is too large the attack takes big steps and possibly overshoots the target, which again yields a poor success rate. Also note that gradient descent is not the most efficient algorithm for solving this kind of optimization problem. However, the aim of this method is to demonstrate potential dangers in data disclosure,.efficiency of the attack is out of the scope.

The gradient, $\nabla E(\mathscr{X}')$, depends on the differentiable properties $P_i, i \in \{1, \ldots, m\}$:

$$\frac{\partial}{\partial \mathscr{X}_i'} E(\mathscr{X}') = \sum_{j=1}^{m} \big(P_i(\mathscr{X}') - P_i(\mathscr{X})\big) \frac{\partial}{\partial \mathscr{X}_i'} P_j(\mathscr{X}') \tag{4.5}$$

If all properties are continuously differentiable, then the gradient is a continuous function in the candidate trajectory.

Recall that not all partial derivatives of the maximum speed property are continuous. The discontinuity happens when more than one segment are equally fast, and are the fastest segments. However, since we defined the derivative to be one in this case, the gradient descent will still change the speed of these segments until they satisfy the known maximal speed.

4.5 Evaluations and Experimental Results

To validate our reconstruction method, we have designed three different tests, and applied them on two datasets of real-world GPS data. One data set consists of routes of school busses in Athens and it represents a more predictable data set since busses will usually follow the same routes. The second data set is obtained in the context of the GeoPKDD project and it consists of the GPS tracks of a set of cars in the city of Milan in Italy. GPS tracks of cars are definitely less predictable since there are many routes that they can follow. In the first test, we let the reconstruction method run for many iterations to see how the success-rate evolves over time. The second test consists of several executions of the reconstruction algorithm on the same dataset, but with a varying number of known trajectories and background information. The aim of the second test is to verify the claim that an attacker can reconstruct a target trajectory with only a few known trajectories. In the third test, we apply Gaussian noise to the released distance data to see how fast the success-rate diminishes in the face of errors.

The first dataset is named *Athens dataset*[20, 1]. This dataset contains 145 trajectories each with 1096 (x, y) sample points. The trajectories are aligned with samples approximately every half minute on 108 different days. This dataset is chosen because of its regularity, which enables us to test our reconstruction algorithm in a near best-case scenario.

The second dataset, *Milan dataset* [2]. This dataset contains 135 trajectories recorded with sample points at irregular intervals over a period of time

of one week. The density of sample points of the Milan dataset is lower than the dataset from Athens. Even though the trajectories in the Milan dataset are not aligned, for the purpose of these tests, we assume that they are. This assumptions only means that we are not working with the original trajectories, but trajectories which follow the same routes, but at different speeds. The Milan dataset is much more complex than the Athens dataset, and is chosen to test our reconstruction algorithm in a scenario which is much more realistic (and relevant) than the Athens dataset.

For the purpose of testing the reconstruction method described in Section 4.4 we implemented a limited version. In the implementation the step-size γ is set to 1, and the implementation does not restart if a local maxima, or saddle point is reached. Furthermore, we assume that the two datasets are aligned, so that we can discard the time component. In all tests in this section we report the success rate as defined in Equation 3.14. We have chosen the smoothness parameter $\alpha = 20$ based on visual impression from several tests.

Even though efficiency is not a primary concern in this work, we remark that it takes approximately 8 minutes to run the reconstruction method with 50 known trajectories from the Athens dataset for 60.000 iterations on a 1.7 GHz laptop.

4.5.1 Success-rate over Time

In the first test, we run the reconstruction method on the Athens dataset for one million iterations to see how the success-rate evolves over time. Figure 3 shows the convergence speed of our reconstruction method. The success-rate is an average value obtained from 5 runs of the reconstruction algorithm on the Athens dataset with 50 known trajectories, where the target trajectory is selected at random in each of the 5 runs. The x-axis shows the number of iterations in log-scale.

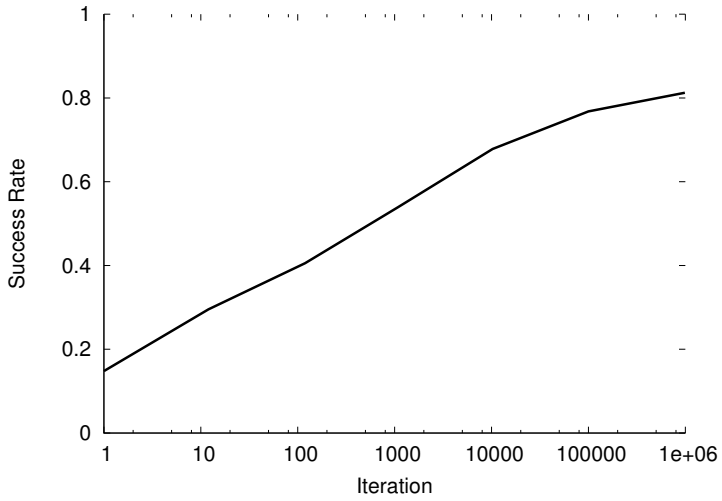


Figure 3: Success-rate vs. number of iterations for the Athens dataset. The x-axis is in log-scale (Average of 5 experiments with 50 known trajectories).

Figure 4 shows the evolution of candidates in one experiment with the Athens dataset and one with the Milan dataset. The test uses 60 known trajectories from the Athens dataset, and 90 known trajectories from the Milan dataset. Notice that a success-rate of 0.6 allows us to determine the general area in which the target trajectory is moving, but not specific streets.

With a success-rate of 0.85 it is possible to identify some, but not all, streets.

4.5.2 Success Rate with Background Information

In the second test, we fix the number of iterations used in the reconstruction to 60.000, and measure the success-rate as a function of the information available to the attacker. We run the reconstruction with a different number of known trajectories, ranging from 10 to 140. We also run the reconstruction both with and without background information about average and maximum speed in the dataset. And finally we run the reconstruction with two different distance measures: Euclidean distance, and average sample distance.

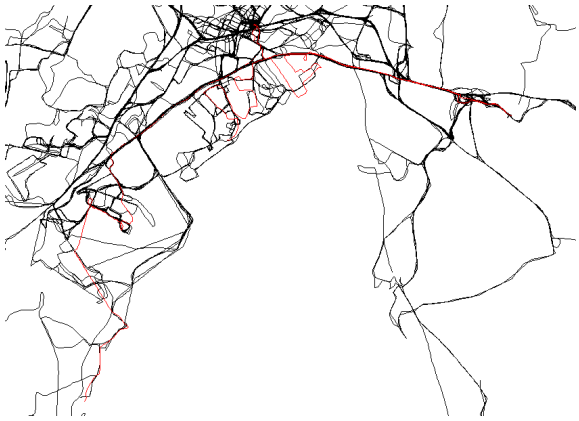
Figure 5 shows the success-rate attainable for different numbers of known trajectories in the Athens dataset. Each sample is the average success-rate of 20 tests each running for 60.000 iterations. Both target and known trajectories are chosen at random in each test. The solid line shows the success rate of the attack, when the attacker only uses the Euclidean distances between the target and the known trajectories as the continuously differentiable properties. The dashed line shows the success rate, when the attacker assumes that the target trajectory moves with an average and maximum speed similar to the average and maximum speed of his known trajectories (see Section3.3). The graph shows that for the case of the Athens dataset, using knowledge about the average speed does not give extra success to the attack. However, Figure 6 shows the same experiment for the Milan dataset, and here it is clear that, for a low number of known trajectories, using knowledge about the average speed gives a success rate up to 0.05 higher (for 20–40 known

trajectories). From the result, we can see that simple background information, such as average and maximum speed, can improve the accuracy of the reconstruction when the quality of the trajectory data is poor (as in the Milan dataset), or in insufficient number of known trajectories are available. However, for good quality trajectory data, the impact of simple background information is not significant. We have only tested speed information, but other kinds of background information may give a higher impact.

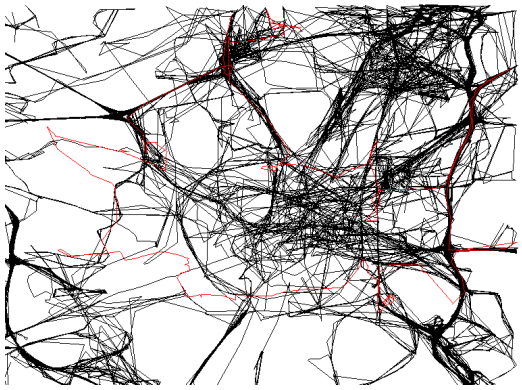
Figure 7 shows the success-rate attainable for different numbers of known trajectories in the Athens dataset when the attacker knows the *average sample distance* to his/her known trajectories. Each sample is the average success-rate of 20 tests each running for 60.000 iterations. Both target and known trajectories are chosen at random in each test. Figure 8 shows the same result for the Milan dataset. The success rate attained from these tests shows that for our attack, knowing the mutual Euclidean distance is stronger than knowing the mutual average sample distances.

4.5.3 Noise

Figure 9 shows the success-rate attainable in the face of errors in the known distances. Independent and identically distributed Gaussian noise with a mean value of 0 has been added to each distance known to the attacker. The Gaussian x-axis of the figure shows the deviation of the noise as a fraction of the average value of the distances. This means that for $x = 1$ approximately 32% of the distances are subject to noise with the same magnitude as the distance itself.



(a) The 60 known trajectories for Athens.



(b) The 90 known trajectories for Milan.

(c) Athens, Success-rate 0.60

(d) Milan, Success-rate 0.60

(e) Athens, Success-rate 0.85

(f) Milan, Success-rate 0.85

Figure 4: Evolution of the candidate trajectory in the Athens and Milan datasets.

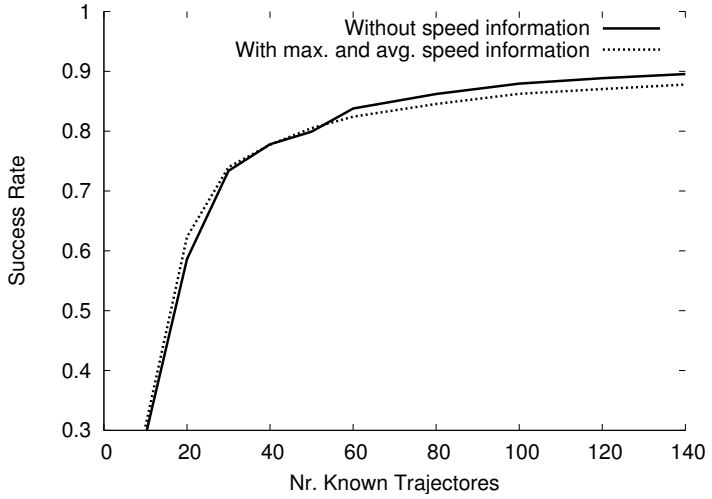


Figure 5: Success-rate vs. number of known trajectories in the Athens dataset with known Euclidean distances. With and without known average and maximum speed.

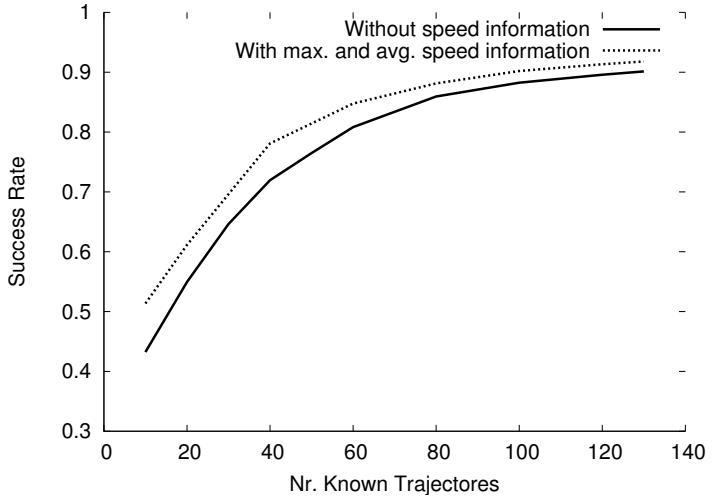


Figure 6: Success-rate vs. number of known trajectories in the Milan dataset with known Euclidean distances. With and without known average and maximum speed.

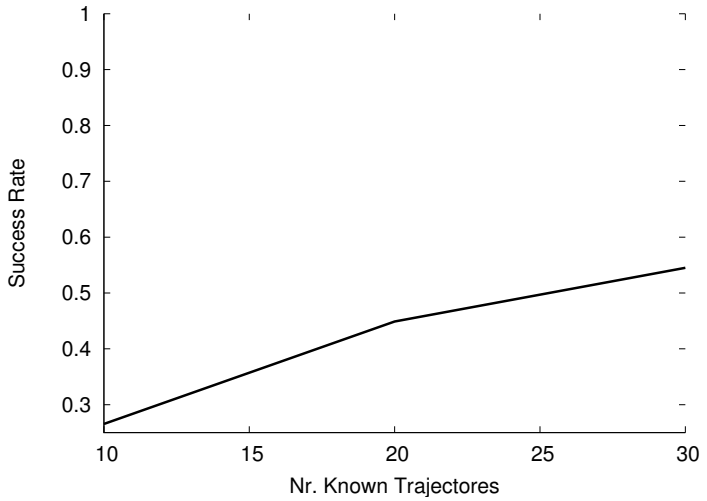


Figure 7: Success-rate vs. number of known trajectories for the Athens dataset with known average sample distance.

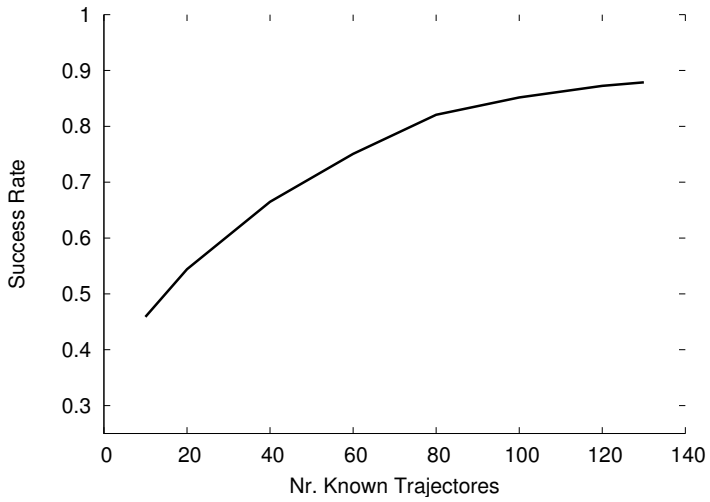


Figure 8: Success-rate vs. number of known trajectories for the Milan dataset with known average sample distance.

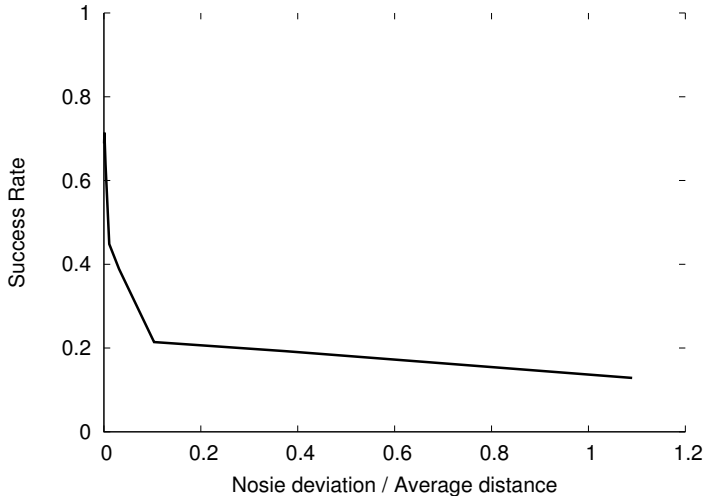


Figure 9: Success-rate for 40 known Euclidean distances subject to noise.

4.6 Discussions

In this chapter, we consider distance preserving data transformations, and assume that the mutual distances of trajectories are released rather than the actual trajectories. We show that, even in such a scenario, the private trajectories can be identified using background information such as known distances and speed limits.

Experiments performed on these real data sets show that unknown private trajectories with 1096 sample points can be reconstructed with an expected success-rate of 0.8 by knowing the distance to only 50 known trajectories while reconstructing the trajectory perfectly with "tri-lateration" would require 2193 known trajectories as studied in [51].

5 Interpolation Based Private Trajectory Discovery

5.1 Introduction

We discussed an attack model in which the adversary uses available background information such as average / maximum speed and those background information played an important role in reconstructing trajectories (see Section 4). In a scenario where the attacker knows some of the trajectories from the dataset together with a released set of mutual distances in-between the trajectories in the dataset, any unknown trajectory from that dataset can be solved. In this chapter, we propose another attack algorithm which utilizes the known trajectories together with the released set of mutual distances. We solve any unknown trajectory mathematically no worse than the actual target trajectory and any other possible solutions. Since the adversary has limited information in terms of known trajectories to perfectly find the target trajectory, s/he may or may not find the exact target trajectory. However s/he can generate all possible solutions which clearly includes the target trajectory itself. Comparing to the first method discussed in Section 4 which finds an arbitrary solution, this method computes all the possible solutions. The set of all possible solutions, so called *candidate trajectories* are utilized to calculate the confidence of the sample points which the target trajectory passes through with high probability. We define the *confidence of an area* as the ratio of candidates passing through a specified sample point or area to all

candidates. The number of candidate trajectories depends on how much information the adversary has. If the adversary has a very limited information, the attack results in a vast number of candidates or even an infinite set of candidate trajectories. We build our method on top of the hyper-lateration technique which is also mentioned in [51]. The number of known trajectories is less than the number of known trajectories in [51]. Because knowing a high number of trajectories and their pairwise distances may be infeasible. Our method works on trajectory data which is assumed to be aligned and have constant sampling rate. This assumption enables us to use interpolation so that some of the points can be imitated by interpolated points hence the number of known trajectories needed is lowered.

This chapter can be outlined as follows:

1. The target trajectory can be solved and the solution is one of the mathematically possible candidates due to the limited information however one can generate various number of candidates which are all identical in terms of distances to the known trajectories so to construct candidate trajectory set.

2. Confidence of an area can be calculated by utilizing the candidate trajectory set to state how certain the target trajectory passes through that area. This analysis is important because this method may not exactly find the target trajectory but using all the solutions generated, it can conclude with high confidence that the target trajectory is passing through a specific area.

5.2 Preliminaries

In this method, we assume that the trajectories follow up paths such as roads and streets. Consecutive sample points should be close to each other which makes interpolation meaningful in this context. Definitions of interpolated trajectory, distance preserving trajectory, candidate trajectory and confidence of an area is discussed below.

Interpolated trajectory is a trajectory which a number of interpolated points are added among its segments.

Distance preserving trajectory is a trajectory which the pairwise distances from the known trajectories is the same as the pairwise distances of the target trajectory from the known trajectories.

Candidate trajectory is the result trajectory of the attack which is a distance preserving trajectory that contains a number of interpolated points.

Confidence of an area is the probability of the target trajectory passing through a certain area. This area is selected to be a sample point, so we discuss about the confidence of a sample point.

5.3 Underlying Methods and Techniques

First of all we don't have enough information to solve unknown variables unless we have enough equations. By re-writing some of the unknown vari-

ables in terms of other points, we reduce the number of unknowns to solve. We use linear interpolation to re-write some of the sample points from the target trajectory. Once we have reduced the number of unknowns, we use a modified hyper-lateration technique to find the target trajectory.

5.3.1 Interpolation Technique

Interpolation is used to create points in between two given points such that they are expected to simulate a real life scenario. Trajectories mostly follow paths such as roads which act like a line when the resolution is high. That is, given a trajectory with considerable number of sample points, it is likely that points follow each other closely. The distance between two sample points in a trajectory depends on the sampling rate and the speed so the points in-between two given points can be predicted by interpolation if the sampling rate of the points are constant. Constant sampling rate is assumed in this thesis (see Section 3.1). It will be fairly easy to interpolate the points since the sampling of the points are constant, so given any two points, so called *main points* and the number of points in-between main points, it is possible to calculate those points in-between, so called *interpolated points.* An interpolated point can be found from the main points using the formula below.

$$I_j(i) = \left(T_{2i} + \frac{j(T_{2i+2} - T_{2i})}{s+1}, T_{2i+1} + \frac{j(T_{2i+3} - T_{2i+1})}{s+1} \right), j = 1, 2, 3...s \tag{5.1}$$

where $I_j(i)$ is the j^{th} interpolated point in the $(i+1)^{th}$ segment, T_{2i} is the $2i^{th}$ coordinate of the trajectory which we add interpolated points. Hence the above formula calculates the x and y values of the $I_j(i)$. Note that T_{2i} and T_{2i+2} corresponds to x values where as T_{2i+1} and T_{2i+3} corresponds to y values of the main points which the interpolated points will lie in-between. For example, in Figure 10, $I_1(0)$ is between main points A and B is the start and end points of the first segment. The coordinates of A and B are T_0, T_1 and T_2, T_3 respectively.

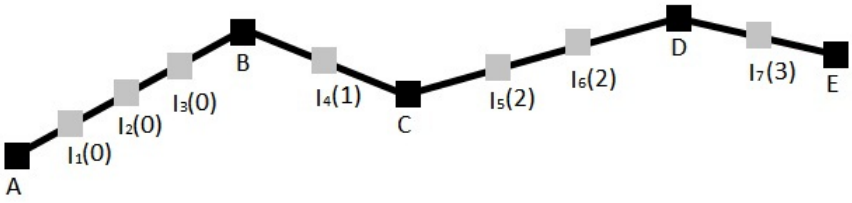


Figure 10: Interpolation

In Figure 10, the black points denotes the main points which are fixed while the grey points denotes the interpolated points. The points I_1, I_2, I_3 are interpolated points in the segment 0 and calculated by Equation 5.1.

5.3.2 Modified Hyper-lateration Technique

Hyper-lateration is a generalization of tri-lateration. It is a simple technique to identify the exact location of a point. In tri-lateration, to identify a point with 2 coordinates namely x and y, 3 other points and their distances to that point are needed. A point in a 2 dimensional plane, R^2, is found by 3 points in R^2. In Figure 11, an example of trilateration is shown. In Figure 11a, the set of points $r1$, $r2$ away from the center of the circles are shown. The points marked with a red circle show the points satisfying the condition so that those points are $r1$, $r2$ away from the center of the circles but we don't know which one is the real solution. This is the case when we do not have sufficient information about the target point so we can find more than one solution as seen in this example. However in Figure 11b, the target point is defined to be $r1$, $r2$, $r3$ away of the center of the 3 circles. This time 3 pairwise distances are enough to *exactly* specify the target trajectory hence the unique solution is found. This example illustrates trilateration, so points are $2d$ and in R^2. As seen from this small example, in R^2 we need $2 + 1 = 3$ points to find the target exactly, anything less than 3 points results in more than one solution.

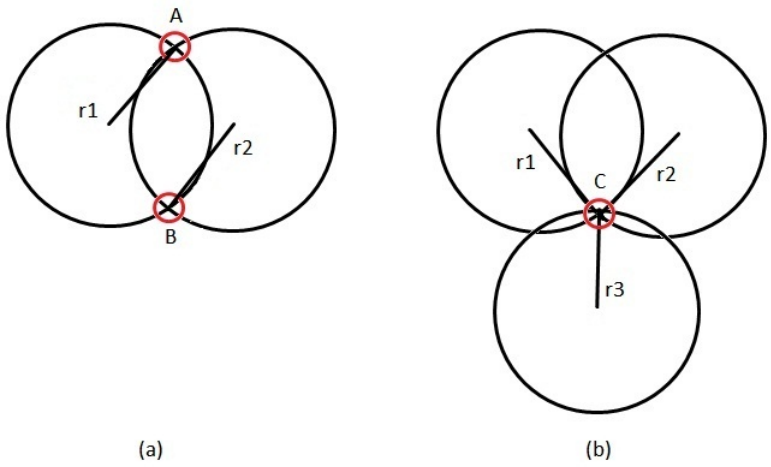


Figure 11: Trilateration

In general, to exactly identify a point in R^{2n}, $2n+1$ points are needed. Trajectories are high dimensional points in space, that is, a trajectory with n sample points ($2n$ coordinates) is a point in R^{2n}. Hence a trajectory can be exactly identified by $2n+1$ points in R^{2n}. The proposed attack methods based on hyper-lateration, such as [51], needs $2n+1$ trajectories to solve an unknown trajectory. This can be done by squaring the known distances (which are in Euclidean distances, so they become Euclidean square distances defined in Section 3.2.1) a system of n quadratic equations is formed such that $\delta_i^2 = ||T - T'||^2 = \sum_{i=0}^{2n-1} |T_i - T_i'|^2$, for $i \in \{1, \ldots, n\}$. Subtracting each of these equations from the first equation yields $n-1$ linear equations. Note that $\mathcal{X}, \mathcal{T}$ denotes the sample point representation, X, T represents the vector representation of the trajectories.

$$\delta_1^2 - \delta_i^2 = \|X - T^1\|^2 - \|X - T^i\|^2 \tag{5.2}$$

$$= \sum_{j=0}^{n-1} \|\mathcal{X}_j - \mathcal{T}_j^1\|^2 - \sum_{j=0}^{n-1} \|\mathcal{X}_j - \mathcal{T}_j^i\|^2 \tag{5.3}$$

Assume that the point $\mathcal{X}_1$ is on the line segment $|\mathcal{X}_0, \mathcal{X}_2|$. If $\mathcal{X}_1$ is an interpolated point denoted as $I_1(0)$, then Equation 5.2 reduce to

$$\delta_1^2 - \delta_i^2 = \|\mathcal{X}_0 - \mathcal{T}_0^1\|^2 - \|\mathcal{X}_0 - \mathcal{T}_0^i\|^2 \tag{5.4}$$

$$+ \|I_1(1) - \mathcal{T}_1^1\|^2 - \|I_1(1) - \mathcal{T}_1^i\|^2 \tag{5.5}$$

$$+ \sum_{j=2}^{n-1} \|\mathcal{X}_j - \mathcal{T}_j^1\|^2 - \sum_{j=2}^{n-1} \|\mathcal{X}_j - \mathcal{T}_j^i\|^2 \tag{5.6}$$

Note that $I_1(0)$ is calculated according to Equation 5.1. According to Equation 5.4, we have the same number of linear equations but one less unknown point to solve because that point is interpolated. This technique can be generalized to further reduce the number of unknown points. To solve the system of linear equations, the number of unknown variables must be equal to the number of equations. Thus the number of interpolated points depends on the number of known trajectories.

According to Equation 5.2, any trajectory satisfying $\delta_1^2 - \delta_i^2 \ \forall i \in \{1, \ldots, n\}$ is a candidate trajectory. Different dissimilarity matrices may have same $\delta_1^2 - \delta_i^2 \ \forall i \in \{1, \ldots, n\}$ but when only *distance differences* are preserved,

candidate trajectories to all different dissimilarity matrices will be the same although the target trajectories are different for each dissimilarity matrix. Thus, a candidate set of distance difference preserving trajectories does not necessarily demonstrate the target trajectory. We can be sure about our candidate trajectories if we preserve the pairwise distances instead of distance differences.

Equation 5.2 preserves distance differences but a distance preserving method can also be derived from the same formula with a new settings. If we have $2n$ known trajectories, $2n-1$ equations are formed with hyperlateration mentioned in Equation 5.2 so that $2n-1$ coordinates can be solved. Assume $2n-1$ coordinates are solved where trajectory has $2n$ coordinates. This means, the last coordinate cannot be solved due to lack of information, but all other coordinates can be expressed in terms of the last coordinate, t_n. We re-write all t_i's in terms of t_n.

$$\sum_{i=1}^{i=2n-1} (2a_i - 2b_i)t_i \; + \; (2a_n - 2b_n)t_n \; = \; d_{bt} - d_{at} + \sum_{i=1}^{i=2n} a_i^2 - b_i^2 \qquad (5.7)$$

$$\sum_{i=1}^{i=2n-1} (2a_i - 2b_i)t_i \; = \; d_{bt} - d_{at} + \left[\sum_{i=1}^{i=2n} a_i^2 - b_i^2\right] \; - \; (2a_n - 2b_n)t_n \qquad (5.8)$$

where d_{xy} represents Euclidean square distance (see Equation 3.5) in-between trajectory X and trajectory Y.

Equation 5.8 can be seen as a matrix operation thus system of equations can easily be solved. This is a matrix operation of $Ax = b$, where x represents all the t_i's, A is the $(2a_i - 2b_i)$ for each t_i and right hand side of the equation is simply the b. To solve all t_i's, multiply both sides with A^{-1} so that it becomes $A^{-1}Ax = A^{-1}b$ hence $x = A^{-1}b$. Here the key point is, the matrix denoted as A must be an *invertible matrix*. A should be invertible so that the system of equations can be solved and a real root can be found. Note that in this system of equations, the root found will be the t_n, the last coordinate of the unknown trajectory. All the t_i's are expressed in terms of t_n so whenever a root is found in this system of equations, the t_n is found automatically. All other coordinates are in terms of t_n so they are found too. To solve Equation 5.8, right hand side of the equation is divided into two main components since it is in the form of $x - y$ where $x = d_{bt} - d_{at} + [\sum_{i=1}^{i=2n} a_i^2 - b_i^2\,]$ and $y = (2a_n - 2b_n)t_n$. Hence we produce two equations to be solved and real t_i's will be in the form of their outputs.

$$\sum_{i=1}^{i=2n-1} (2a_i - 2b_i)e_i \;=\; d_{bt} - d_{at} + \left[\sum_{i=1}^{i=2n} a_i^2 - b_i^2\right] \tag{5.9}$$

$$\sum_{i=1}^{i=2n-1} (2a_i - 2b_i)f_i \;=\; (2a_n - 2b_n)t_n \tag{5.10}$$

Equations 5.9 and 5.10 are in the form of $Ae = x$ and $Af = y$. Calculating Equation 5.9 and Equation 5.10 as separate matrix operations yields two different solutions to be combined to find real t_n. Solving this system of equations as matrix operation is explained above. When both Equation 5.9

and Equation 5.10 and solved, Equation 5.9 yields e_i , $\forall\, i \in \{1, 2, ..., 2n-1\}$ and Equation 5.10 yields f_i , $\forall\, i \in \{1, 2, ..., 2n-1\}$. Since the right hand side of the Equation 5.8 is in form of $x - y$ the real t_i's will be in the form of

$$t_i = e_i - f_i t_n,\ \forall\, i \in \{1, 2, ..., 2n-1\} \tag{5.11}$$

All t_i's according to Equation 5.11 are constructed. Since all t_i's are in terms of t_n that we want to find, we can plug these t_i's into the distance calculation (see Section 3.2) between any known trajectory, k and the unknown trajectory. Distance between k and t is as follows:

$$\sum_{i=1}^{i=2n}(k_i - t_i)^2 = d_{kt} \ \rightarrow\ \sum_{i=1}^{i=2n}(k_i - t_i)^2 - d_{kt} = 0 \tag{5.12}$$

Plugging Equation 5.11 into Equation 5.12 results in

$$\left[\sum_{i=1}^{i=2n-1} f_i^2 t_n^2 + (2k_i f_i - 2e_i f_i)t_n + (k_i - e_i)^2\right] + (k_n^2 - 2k_n t_n + t_n^2)\ - d_{kt} = 0 \tag{5.13}$$

The roots of Equation 5.13 can be solved by solving quadratic equations. Solving quadratic equations and finding real roots are explained in [5]. Since Equation 5.13 is a quadratic equation in the form of $ax^2 + bx + c = 0$, the real root will be t_n. Since t_n is found, the coordinates, t_i's , that are expressed in terms of t_n's are also found.

The real roots, which will be t_n can be found using the formula below.

$$x_1 = \frac{-b - sgn(b)\sqrt{b^2 - 4ac}}{2a} \tag{5.14}$$

$$x_2 = \frac{c}{ax_1} \tag{5.15}$$

where *sgn* denotes sign function which returns the sign of the given argument (i.e if it is positive then it returns 1, else it returns -1).

5.4 Our Method

5.4.1 The Method and Confidence of an Area

Our approach is based on hyper-lateration (see Section 5.3.2) together with interpolation (see Section 5.3.1). The adversary, has a set of trajectories, say k trajectories, with s sample points. S/he computes $k-1$ equations by using the Equation 5.2. According to Equation 5.2, distance differences are preserved between the candidate trajectory and the known trajectories regardless of how close the resulting trajectory is to the target trajectory. If $k = s$, hyper-lateration will result the exact unknown trajectory and we are done. When $k < s$, it can only solve up to $k-1$ coordinates. Due to lack of information, we can not find an exact solution. Under the assumption of constant sampling rate mentioned in Section 3.1, we use the interpolation technique (see Section 5.3.1) for the rest of the points. This solvable $k-1$

coordinates are called *main coordinates*. Two main coordinates (x and y coordinates) forms a *main point*. We re-write the interpolated points in terms of their main points hence when they are solved, candidate trajectory is automatically formed. When $k-1$ coordinates are solved, there are $s-k-1$ coordinates needed to be interpolated. Those interpolated points are distributed randomly among the segments of the candidate trajectory. Those segments are formed by the main points of the candidate trajectory. In each run of finding candidates (i.e each run of the attack), different number of interpolated points are added to the segments hence a different candidate trajectory is formed. When $k = s$, target trajectory is exactly found. Otherwise, we measure the success of the found trajectory because there are many candidates which satisfies all the properties. As the number of known trajectories by the adversary decreases, the number of candidate trajectories increases. If the information is very limited, so that the number of coordinates to be interpolated is very high compared to the number of coordinates that can be solved, the attack even becomes senseless. Success rate is measured as in Section 3.4.2. Our method gives the best possible results in terms of distances because pairwise distances are preserved so mathematically any candidate trajectory found with this method is no less likely to be the target trajectory than any other candidate trajectory.

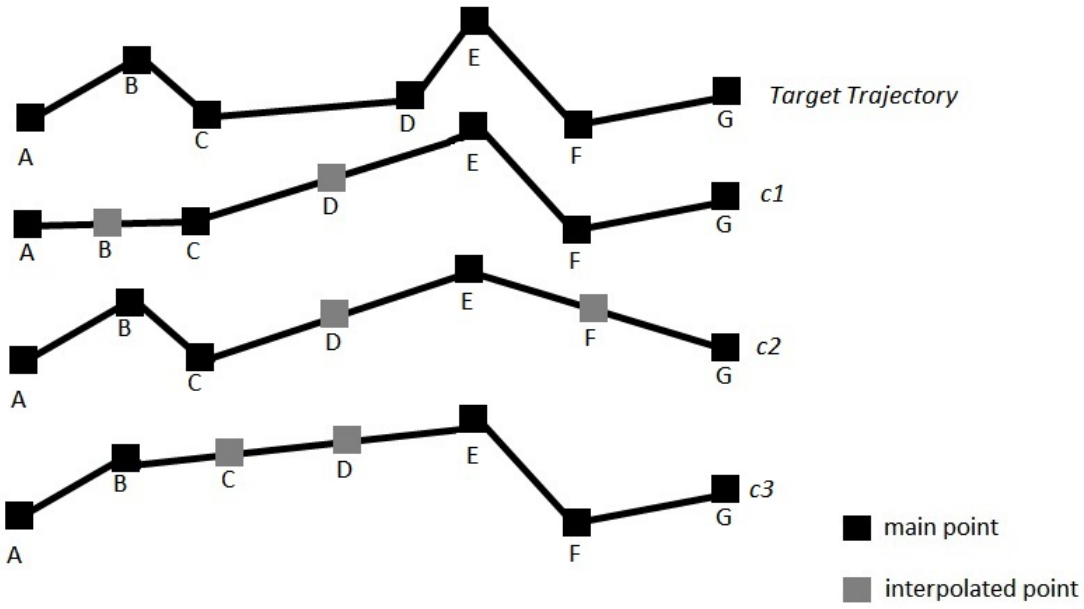


Figure 12: Candidate Generation

When $k < s$ there are more than one solution to the equation system where each solution is a candidate. Each solution is mathematically identical to each other within given properties, so the known distances, which is the only information the adversary have. In Figure 12, candidate trajectory generation is shown. With the limited information we have, we can solve up to $k-1$ coordinates ($(k-1)/2$ main points) which are the denoted by the black points. In between two main points, we have a segment where we can put a number of interpolated points which are denoted with grey points. Every segment can have a different number of interpolated points depending on the distribution of the interpolated points. We distribute a random number

of interpolated points to each segment so the total number of sample points will be equal to the total number of sample points of the other trajectories in the dataset. This way, candidate trajectory has same number of sample points as the target trajectory. As seen in Figure 12, when the number of interpolated points of the segment changes, a new candidate trajectory originates. Each candidate is equally likely to be the target trajectory even though they may not be the target trajectory. Because all candidates satisfies the known distances property. Distances from those candidates to the known trajectories are equal to each other. Moreover the distances between a candidate and the known trajectories are equal to the distances from the target trajectory to known trajectories.

Candidate trajectories, $c1, c2, c3$ in Figure 12 are all indistinguishable in terms of pairwise distances and visually different trajectories. This set of candidate trajectories is called *candidate trajectory set.* We assume that the candidate set eventually contain the target trajectory when all possible solutions are generated. While each candidate in this set is a possible target trajectory, the area covered by these candidates is possibly the area where the actual target trajectory passes through. We do not only find a good solution but also compute the confidence on the solutions. *Confidence of an area* is the ratio of the number of candidate trajectories which passes through a specific area, or a sub-trajectory, or a sample point in the trajectory (say l^{th} sample point) to the cardinality of the candidate trajectory set. Confidence of an area around the sample point l , denoted as s_l is calculated as follows:

$$C_l = \frac{\#\, of\ candidates\, passing\, s_l}{cardinality\, of\ candidate\, trajectory\, set} \tag{5.16}$$

For instance, in Figure 13, assume that we have 5 candidates, namely $c1, c2, c3, c4, c5$, for a target trajectory, t, each of which has different success rate according to Success Rate 2 in 3.14. Consider the area $1m^2$ around s_l, which is marked in a green rectangle denoted as A_l. After examining the candidates, $c1, c2, c3, c4$ are passing through A_l while $c5$ is not passing through so 4 out of 5 candidates pass through A_l, hence we conclude that the target trajectory, t, passes through A_l with confidence $4/5 = 80\%$. This section aims to find the target trajectory and discuss the confidence of areas that are possibly to be used by the target trajectory.

The power of this method is based on how successful the interpolation is done over the trajectory. If the target trajectory is following up a linear path, so the interpolation works great and we may have very high success rates. At least one candidate trajectory will be close to the target trajectory. Otherwise, the success rate may be very low or even becomes 0.

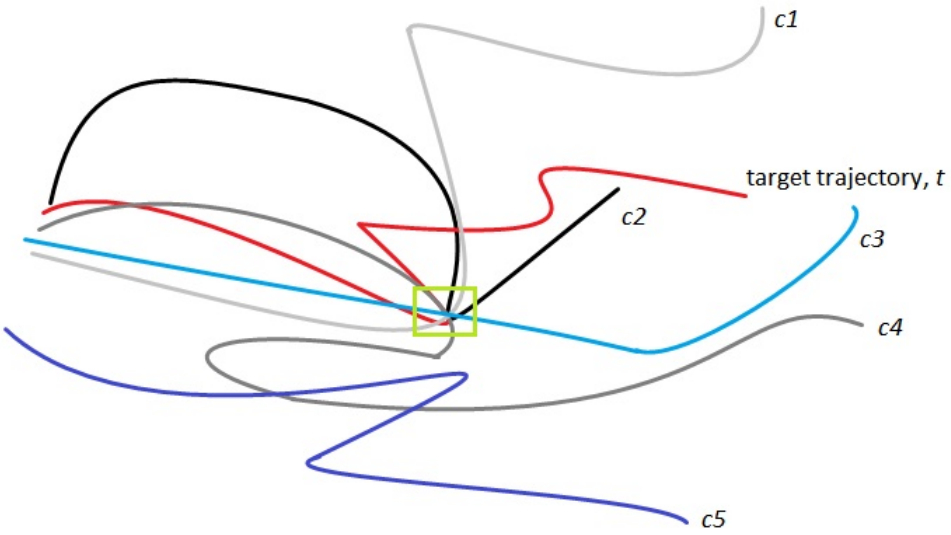


Figure 13: Confidence of an Area

5.4.2 Evaluations and Experimental Results

This method is tested on a synthetic data that we created. Our aim is to show that this method works and present the confidence of the areas where the unknown trajectory likely to pass through. Our experiments show that, all the candidate trajectories we found have the same distances given in the dissimilarity matrix. The dissimilarity matrix contains the pairwise distances between the known trajectories and the target trajectory. This makes them *mathematically indistinguishable* from each other and from the target trajectory. In Figure 14, success rate versus the number of known trajectories is shown. According to *Success Rate 2* defined in 3.14, average

success rate of the whole attack is $0,53$ when the $\alpha = 20$. Success rate was calculated, for a fixed target trajectory which has 20 coordinates.

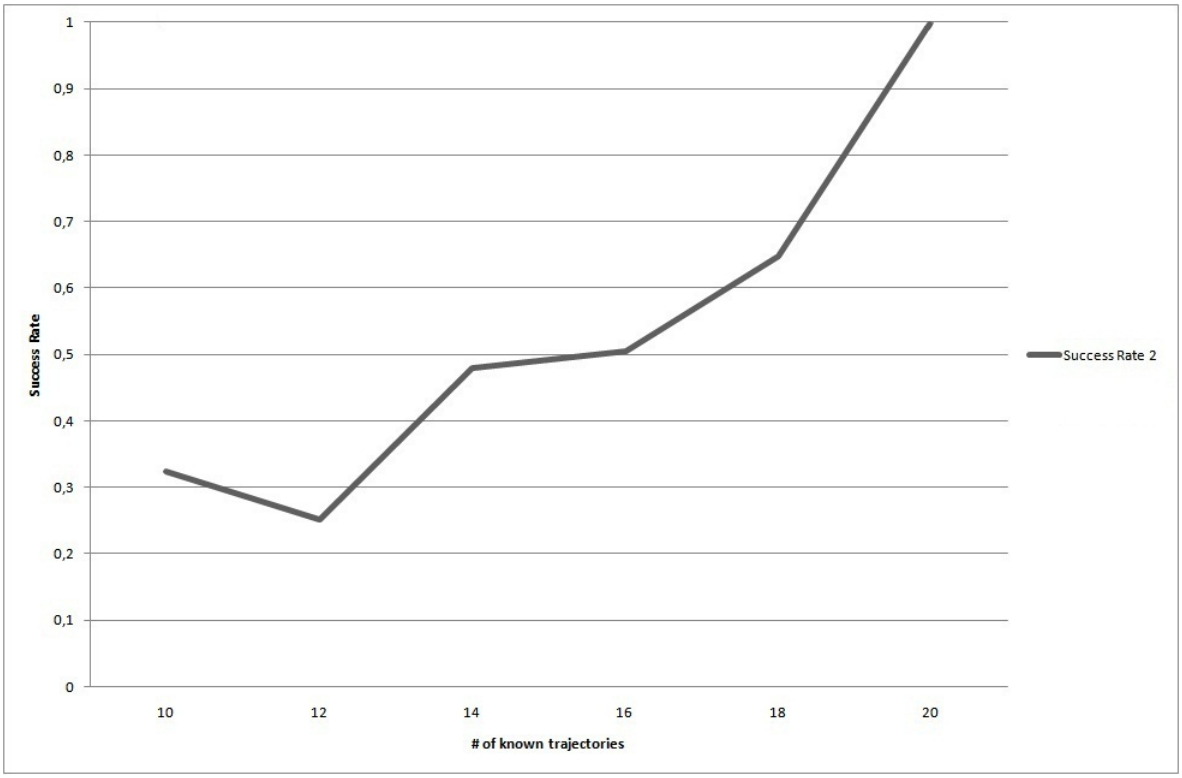


Figure 14: Success Rate vs no of known trajectories

5.5 Discussions

In this section, we present a method for finding unknown trajectories with a limited information such as a very few known trajectories and their pairwise distances to other trajectories. Our method finds candidates which is the best possible candidates and they are mathematically same and no less likely than the real target trajectory because the pairwise distances among them

are preserved. Using hyper-lateration technique needs $2n + 1$ trajectories to be known beforehand while our method can decrease that number up to 40% depending how linear the target trajectory is. Linear structure among the sub-parts of the trajectories is the key part in our method because we fill those missing information with the interpolated points so that if the unknown trajectory has a linear sub-structures, the interpolated points that we placed among the candidate will conform to the real target and the rest of the points are solved because we had those information from the system of equations. In other words, assuming we have 50 trajectories, we calculate up to 49 coordinates while unknown target trajectory has 90 coordinates, we place $90 - 49 = 41$ coordinates which are produced by interpolation technique and distributed randomly across the sub-segments of the candidate that we create and treat as the unknown trajectory. Due to the time constraints, calculating confidence of the sample points or an area is leaved as a future work which we are currently working on. Our method is sensitive to the noise can be presented in dissimilarity matrix because a change in the distances will result in candidates which are wrongly placed but still preserve the pairwise distances which contains noise. Another future work can be strengthen this method against noise in the dissimilarity matrix.

6 Comparisons

In this thesis, we presented two attack methods which reveal private trajectories using background information. The ultimate aim in both of the methods is to show privacy leaks. However these methods differ not only in the way of computations but also in their settings.

In the first method presented in Section 4, the aim is to recover the private trajectory. This method exhausts various kinds of background information and results in one arbitrary solution with a high success rate possible. This solution may not be the real target trajectory but due to limited information, it is the best solution so far and satisfy all available information about the target trajectory. Other possible solutions are *not* calculated in this method. This is a heuristic based method and exhausts the given background information. It also works when there is noise in the dissimilarity matrix and can even handle complex real world datasets like the Milan dataset. Moreover it can use almost any kind of background information.

In the second method discussed in Section 5, the aim is to recover the private trajectory by calculating all possible candidate trajectories satisfying the given information and further discuss about confidence of the selected areas. This method exhausts only the dissimilarity matrix and a set of known trajectories. Using the available but limited information to the adversary, possible solutions (i.e the candidates) are calculated and a candidate trajectory set is formed. The real target trajectory may or may not be in this set but eventually if all the candidates are generated, then the target trajectory

will be in this set. Calculating all the candidates may be infeasible if the information is very limited. Although we may not find the target trajectory, using the possible candidate trajectories, we can discuss about the areas where the target trajectory passes through. This method is based on linear interpolation which works better if the sample points in the trajectories are close like in the Athens dataset. Moreover hyper-lateration is sensitive to noise and we do not expect high success rate in noisy and complex datasets like the Milan dataset. This method needs more available information compared to first method to yield good success rates.

In both of the methods, the key point determining the success rate is the available information to the adversary. Exhausting limited information yields more than one solution because theoretically finding the exact solution needs $2n + 1$ known trajectories. When the available information is high, success rates in both attack methods are high too.

7 Conclusions and Future Work

In this thesis, we studied privacy leaks that may occur in case of distance preserving data transformations on spatio-temporal datasets, especially in trajectory publishing. We presented two methods, which are based on available background information, and a dissimilarity matrix to show that sharing dissimilarity matrices for data mining is not exactly a privacy preserving solution. Background information is in terms of known distances, average and maximum speed. The only given information is the dissimilarity matrix which is released for data mining purposes. Speed limit is not given information though adversary can easily learn it because it is a set of rules that defines the maximum speed on different types of roads and the average speed can also be computed according to traffic density of the roads. In fact, traffic density applications are available on the internet for many cities around the world so the average speed can be obtained from such information sources. Our methods demonstrate that with limited information, the adversary can precisely find the target trajectory which is clearly a privacy leak in dissimilarity matrix release.

In the first method presented in Section 4, our contribution is to demonstrate the privacy leak in dissimilarity matrices with a very limited information compared to hyper-lateration attack discussed in [51]. We also utilized other properties of trajectories such as average and maximum speed and show their effect on the success rate of the attack. We use the speed limit as background information, but the attack model we propose is general enough so that any kind of background information about trajectories with continu-

ous properties could be the input. Our method is optimal in the sense that it will eventually find a candidate which exhausts all the information available about the unknown trajectory. The experiments show that, with known trajectories we can find very precise candidates with a high success rate.

In the second method presented in Section 5, we demonstrate an advanced version of hyper-lateration attack which requires less information than the original one. Although $2n + 1$ known trajectories are needed for a perfect attack, we are still able to find privacy leaks with limited information. In case the number of known trajectories is smaller than $2n+1$, we use an interpolation technique to infer some of the points in terms of the known points. Contribution of this method is not limited to privacy leak demonstration but we also discuss the confidence of an area that target trajectory is possibly passes through. Since the background information is limited, there can be a number of candidate trajectories. The area that the real target trajectory passes can be discovered from the candidates by looking at the intersections of their surrounding areas.

The methods presented in this thesis show that privacy-preserving spatio-temporal data mining using data transformation techniques may not be an appropriate solution. This shows that privacy in spatio-temporal data mining context needs further research against background information-based attacks. In this thesis, we address the privacy leaks in trajectory data publishing and address the weakness of dissimilarity matrix release in terms of privacy. As a future work, we plan to study the effect of noise. We are currently looking at confidence of an area discussed in the second method. We believe that

revealing location information of individuals is as important as revealing the whole target trajectory. In the first method, we aim to reveal the trajectory without considering the sensitive locations that the person may pass through while in the second method we address the need of a measure to calculate how likely the person passes through a selected location.

References

[1] Athens dataset. http://www.rtreeportal.org/.

[2] Geopkdd. http://www.geopkdd.eu.

[3] Global positioning system. http://en.wikipedia.org/wiki/Global_Positioning_System.

[4] Privacy. http://en.wikipedia.org/wiki/Privacy.

[5] Solving quadratic equations. http://en.wikipedia.org/wiki/Quadratic_equation.

[6] *Location Privacy in Mobile Systems: a Personalized Anonymization Model*, 2005. Proceedings of the 25th International Conference on Distributed Computing Systems.

[7] First interdisciplinary workshop on mobility, data mining and privacy, rome, italy. http://wiki.kdubiq.org/mobileDMprivacyWorkshop/, February 2008.

[8] O. Abul and F. Bonchi. Never walk alone: Uncertainty for anonymity in moving objects databases. In *The 24th International Conference on Data Engineering (ICDE 2008)*, 2008.

[9] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the ACM SIGMOD Conference on Management of Data*, pages 439–450. ACM Press, May 2000.

[10] C. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, and P. Samarati. Location privacy protection through obfuscation-based techniques. pages 47–60. 2007.

[11] J. Baugh and J. Guo. Location privacy in mobile computing environments. pages 936–945. 2006.

[12] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. *Pervasive Computing, IEEE*, 2(1):46–55, 2003.

[13] A. R. Beresford and F. Stajano. Mix zones: user privacy in location-aware services. In *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*, pages 127–131, 2004.

[14] C. Bettini, S. Mascetti, X. S. Wang, and S. Jajodia. Anonymity in location-based services: Towards a general framework. In *MDM*, pages 69–76, 2007.

[15] C. Bettini, X. S. Wang, and S. Jajodia. Protecting privacy against location-based personal identification. LNCS 3674, pages 185–199, 2005.

[16] J. V. C. Clifton, M. Kantarcýoðlu, X. Lin, and M. Y. Zhu. Tools for privacy preserving data mining. pages 28–34, 2004. ACM SIGKDD Explorations.

[17] C. Clifton, M. Kantarcioglu, and J. Vaidya. Defining privacy for data mining. 2002.

[18] C. Clifton, D. Mulligan, and R. Ramakrishnan. Data mining and privacy: An overview. pages 191–208. 2006.

[19] J. Domingo-Ferrer and V. Torra. Privacy in data mining. *Data Mining and Knowledge Discovery*, 11(2):117–119, September 2005.

[20] E. Frentzos, K. Gratsias, N. Pelekis, and Y. Theodoridis. Nearest neighbor search on moving object trajectories. In *SSTD05: Advances in Spatial and Temporal Databases*, pages 328–345, 2005.

[21] E. Frentzos, K. Gratsias, and Y. Theodoridis. Index-based most similar trajectory search. In *ICDE*, pages 816–825, 2007.

[22] S. L. Garfinkel, A. Juels, and R. Pappu. Rfid privacy: an overview of problems and proposed solutions. *Security & Privacy Magazine, IEEE*, 3(3):34–43, 2005.

[23] B. Gedik and L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE Transactions on Mobile Computing*, 7(1):1–18, January 2008.

[24] F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli. Mining sequences with temporal annotations. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 593–597. ACM, 2006.

[25] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 330–339. ACM, 2007.

[26] F. Giannotti and D. Pedreschi. *Mobility, Data Mining and Privacy: Geographic Knowledge Discovery*. Springer Publishing Company, Incorporated, 2008.

[27] M. Gruteser and X. Liu. Protecting privacy in continuous location-tracking applications. 2004. IEEE Security and Privacy.

[28] D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. *Bulletin of Mathematical Biology*, 55(1):141–154, January 1993.

[29] J. Han and M. Kamber. *Data Mining, Second Edition, Second Edition : Concepts and Techniques (The Morgan Kaufmann Series in Data Management Systems) (The Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann, January 2006.

[30] A. Inan. Privacy preserving distributed spatio-temporal data mining. Master's thesis, Sabancı University, 2006.

[31] A. Inan and Y. Saygin. Privacy preserving spatio-temporal clustering on horizontally partitioned data. In *DaWaK*, pages 459–468, 2006.

[32] A. Inan, Y. Saygin, E. Savas, A. A. Hintoglu, and A. Levi. Privacy preserving clustering on horizontally partitioned data. In *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*, page 95, 2006.

[33] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 279–288, New York, NY, USA, 2002. ACM Press.

[34] M. Kantarcioglu, J. Jin, and C. Clifton. When do data mining results violate privacy? In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 599–604, New York, NY, USA, 2004. ACM Press.

[35] E. Kaplan, T. B. Pedersen, E. Savaş, and Y. Saygın. Privacy risks in trajectory data publishing: Reconstructing private trajectories from continuous properties. In *KES 2008: Knowledge-Based Intelligent Information and Engineering Systems*, volume 5178 of *Lecture Notes in Computer Science*, pages 642–649. Springer-Verlag, 2008.

[36] E. Kaplan, T. B. Pedersen, E. Savaş, and Y. Saygın. Discovering private trajectories using background information. In *DKE Special Issue*, 2009.

[37] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *ICDM*, pages 99–106, 2003.

[38] J. Lee, J. Han, and K. Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 593–604. ACM, 2007.

[39] X. Li, J. Han, J.-G. Lee, and H. Gonzalez. Traffic density-based discovery of hot routes in road networks. In *SSTD 2007: 10th International Symposium on Advances in Spatial and Temporal Databases*, Lecture Notes in Computer Science, pages 441–459. Springer, 2007.

[40] K. Liu, H. Kargupta, and J. Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Trans. Knowl. Data Eng.*, 18(1):92–106, 2006.

[41] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new casper: A privacy-aware location-based database server. In *ICDE*, pages 1499–1500, 2007.

[42] S. Mukherjee, M. Banerjee, Z. Chen, and A. Gangopadhyay. A privacy preserving technique for distance-based classification with worst case privacy guarantees. *Data Knowl. Eng.*, 66(2):264–288, 2008.

[43] M. Nanni. *Clustering Methods for Spatio-Temporal Data.* PhD thesis, University of Pisa, 2002.

[44] M. Nanni and D. Pedreschi. Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, 27(3):267–289, November 2006.

[45] C. J. Needham and R. D. Boyle. Performance evaluation metrics and statistics for positional tracker evaluation. In *Third International Conference on Computer Vision Systems, ICVS 2003*, pages 278–289, 2003.

[46] E. Nergiz, M. Atzori, and Y. Saygin. Towards trajectory anonymization: a generalization-based approach. In *In Proceedings of ACM GIS Workshop on Security and Privacy in GIS and LBS*, CA, USA, 2008.

[47] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13:1010–1027, 2001.

[48] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). *PODS*, page 188, 1998.

[49] Y. Saygın, V. S. Verykios, and C. Clifton. Using unknowns to prevent discovery of association rules. *SIGMOD Record*, 30(4):45–54, 2001.

[50] M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *The 9th International Conference on Mobile Data Management (MDM'08)*, pages 65–72, 2008. The 9th International Conference on Mobile Data Management (MDM'08).

[51] E. O. Turgay, T. B. Pedersen, Y. Saygın, E. Savaş, and A. Levi. Disclosure risks of distance preserving data transformations. In *SSDBM 2008: Scientific and Statistical Database Management Conference*, 2008.

[52] J. Vaidya and C. Clifton. Privacy-preserving data mining: why, how, and when. *Security & Privacy, IEEE*, 2(6):19–27, 2004.

[53] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *SIGMOD Rec.*, 33(1):50–57, 2004.

[54] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 673–684, 2002.