Uncertainty for Anonymity and 2-Dimensional Range Query Distortion

Spyros Sioutas[1], Emmanouil Magkos[1], Ioannis Karydis[1], and Vassilios S. Verykios[2]

[1] Department of Informatics, Ionian University, Corfu, Greece
{sioutas,emagos,karydis}@ionio.gr
[2] Department of Computer and Communication Engineering, University of Thessaly, Greece
verykios@inf.uth.gr

Abstract. In this work, we study the problem of anonymity-preserving data publishing in moving objects databases. In particular, the trajectory of a mobile user on the plane is no longer a polyline in a two-dimensional space, instead it is a two-dimensional surface: we know that the trajectory of the mobile user is within this surface, but we do not know exactly where. We transform the surface's boundary poly-lines to dual points and we focus on the information distortion introduced by this space translation. We develop a set of efficient spatio-temporal access methods and we experimentally measure the impact of information distortion by comparing the performance results of the same spatio-temporal range queries executed on the original database and on the anonymized one.

Keywords: Uncertainty, Privacy, Anonymity, Moving Objects Databases.

1 Introduction

The technological advances in sensors and wireless communications have made possible the offering of high accuracy in location tracking at a low cost [3], [4], [8]. The increased location accuracy gave rise to a series of location-based applications that exploit positional data to offer high-end services to their subscribers [5]. We consider a population of mobile users who are supported by some telecommunication infrastructure, owned by a telecom operator. Every user periodically transmits through his/her mobile device a location update to some traffic monitoring system residing in a trusted server of the telecom operator. The transformation of the exact user location to a spatiotemporal area is achieved through the use of k-anonymity. The k-anonymity principle for relational data [15], [16] requires that each record in a given dataset is indistinguishable from at least $k-1$ other records with respect to a certain set of identifying variables, collectively known as the "quasi-identifier". The k-anonymity principle requires that the spatiotemporal area that is generated by the trusted server from the exact location of the mobile user is such that the identity of the user cannot be disclosed with a probability that is larger than $1/k$, among $k-1$ other users.

J. Domingo-Ferrer and E. Magkos (Eds.): PSD 2010, LNCS 6344, pp. 85–96, 2010.

This trusted server is requested to efficiently answer Range Queries (RQs) of mobile users moving on the plane.

In our privacy model we assume an attacker who has knowledge of (i) the frequent movement behavior of all the users in the system, computed by the trusted server as part of its functionality, (ii) the anonymized location updates of the users, as received and anonymized by the trusted server, and (iii) the algorithms used by the trusted server to support user privacy. Our solution is capable of ensuring the privacy of the users, even in the case that all this diverse knowledge is at the disposal of the attacker. k-anonymity is essential to protect the privacy of the users, starting from the point of request for a RQ service and continuing for as long as the requested service withstands completion. As part of our framework, we deliver the type of k-trajectory anonymity [6] that identifies $k-1$ users that are close to the requester at the time of request and thus could have issued the request. This includes a minimum circular spatial area A_{min} around the requester, where the participants of the anonymity set should be searched for so that the user is adequately covered up. The proposed framework deals with the event of failure in the provision of k-anonymity, in the case where the number of participants inside this minimum spatial area is less than k-1. In this case, the trusted server postpones the servicing of the user request for a small period of time. After that, if the anonymization process fails again, the requester is protected by blocking the servicing of the request. The proposed privacy framework relies on a user privacy profile that stores the necessary information related to his/her privacy requirements. This includes (i) the preferred value of k (in k-anonymity) for each requested RQ service, (ii) the minimum circular spatial area A_{min}, around the requester, where the participants of the anonymity set should be searched for so that the user is adequately covered up. This threshold defines the minimum extent of the spatial area that must replace the real location of the user, in the anonymized request.

Based on the proposed privacy model we implement a framework that uses the Spatial extensions of MySql 5.x to offer privacy in RQ services. This type of queries focuses on the problem of indexing mobile users in two dimensions and efficiently answering range queries over the users locations. This problem is motivated by a set of real-life applications such as intelligent transportation systems, cellular communications, and meteorology monitoring. There are two basic approaches used when trying to handle this problem; those that deal with discrete and those that deal with continuous movements. In a discrete environment the problem of dealing with a set of moving objects can be considered to be equivalent to a sequence of database snapshots of the object positions/extents taken at time instants $t_1 < t_2 < \ldots$, with each time instant denoting the moment where a change took place. From this point of view, the indexing problems in such environments can be dealt with by suitably extending indexing techniques from the area of temporal [17] or/and spatial databases [7]; in [12] it is elegantly exposed how these indexing techniques can be generalized to handle efficiently queries in a discrete spatiotemporal environment. When considering continuous movements there exists a plethora of efficient data structures [9,11,13,14,18]. The

common thrust behind these indexing structures lies in the idea of abstracting each object's position as a continuous function $f(t)$ of time and updating the database whenever the function parameters change; accordingly an object is modeled as a pair consisted of its extent at a reference time (design parameter) and of its motion vector. One categorization of the aforementioned structures is according to the family of the underlying access method used. In particular, there are approaches based either on R-trees or on Quad-trees. On the other hand, these structures can be also partitioned into (a) those that are based on geometric duality and represent the stored objects in the dual space [11,14] and (b) those that leave the original representation intact by indexing data in their native n-d space [2,13,18]. The *geometric duality transformation* is a tool heavily used in the Computational Geometry literature, which maps hyper-planes in R^n to points and vice-versa.

In this work, we study the problem of anonymity-preserving data publishing in moving objects databases. In particular, the trajectory of a mobile user on the plane is no longer a polyline in a two-dimensional space, instead it is a two-dimensional surface: we know that the trajectory of the mobile user is within this surface, but we do not know exactly where. We transform the surface's boundary poly-lines to dual points [11,13] and we focus on the information distortion introduced by this space translation. We develop a set of efficient spatio-temporal access methods and, we experimentally measure the impact of information distortion by comparing the performance results of the same spatio-temporal range queries executed on the original database and on the anonymized one.

In Section 2 we give a formal description of the problem. In Section 3 we present the method of transforming the trajectory poly-lines to two-dimensional surfaces. In Section 4 we present the duality transformation of surface's boundary poly-lines and we focus on the information distortion introduced by this space translation. Section 5 presents an extended experimental evaluation and section 6 concludes the paper.

2 Definitions and Problem Description

We consider a database that records the position of moving objects in two dimensions on a finite terrain. We assume that objects move with a constant velocity vector starting from a specific location at a specific time instant. Thus, we can calculate the future object position, provided that its motion characteristics remain the same. Velocities are bounded by $[u_{min}, u_{max}]$. Objects update their motion information, when their speed or direction changes. The system is dynamic, i.e. objects may be deleted or new objects may be inserted. Let $P_z(t_0) = [x_0, y_0]$ be the initial position at time t_0 of object z. If object z starts moving at time $t > t_0$, its position will be $P_z(t) = [x(t), y(t)] = [x_0 + u_x(t - t_0), y_0 + u_y(t - t_0)]$, where $U = (u_x, u_y)$ is its velocity vector. For example, in Figure 1 the lines depict the objects trajectories on the (t, y) plane. We would like to answer queries of the form: "Report the objects located inside the rectangle $[x_{1_q}, x_{2_q}] \times [y_{1_q}, y_{2_q}]$ at the time instants between t_{1_q} and t_{2_q} (where $t_{now} \leq t_{1_q} \leq t_{2_q}$), given the current motion information of all objects".

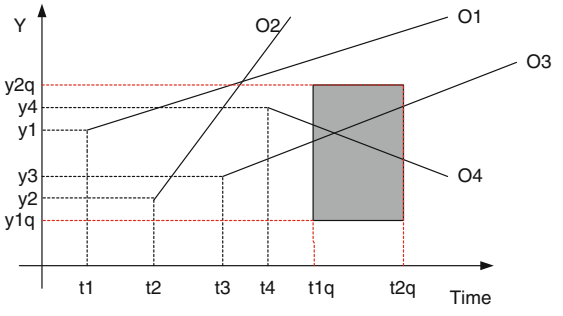

Fig. 1. Trajectories and query in (t, y) plane

3 Trajectory Poly-Lines and Two-Dimensional Surfaces

For every mobile user, we calculate a circular range query with center its current 2-D location and radious a given value R defined by a minimum circular spatial area A_{min}. If this circular spatial area includes at least $k-1$ other neighbours, then the mobile user is adequately covered up. Otherwise, if the number of participants inside this minimum spatial area is less than k-1, the trusted server postpones the servicing of the user request for a small period of time. After that, if the anonymization process fails again, the requester is protected by blocking the servicing of the request. As a result, consecutive circular spatial areas construct a 2-D buffer defined by its upper and lower boundary poly-lines y' and y'' respectively, which anonymize the original trajectory y of mobile user A (see figure 2). By using the Spatial extensions of MySql 5.x we can create each mobile user as 2-dimensional point as follows:

```
CREATE TABLE Points (
name VARCHAR(20) PRIMARY KEY,
location Point NOT NULL,
description VARCHAR(200),
SPATIAL INDEX(location)
);
```

In order to obtain points in a circular area as a counted result ordered by distance from the center of the selection area, we write:

```
SELECT COUNT(name), AsText(location), SQRT(POW( ABS( X(location)
- X(@center)), 2) + POW( ABS(Y(location) - Y(@center)), 2 )) AS distance
FROM Points
WHERE Intersects( location, GeomFromText(@bbox) )
AND SQRT(POW( ABS( X(location) - X(@center)), 2) +
POW( ABS(Y(location) - Y(@center)), 2 )) ≤ @R
ORDER BY distance;
```

If the result returned is less than k-1, the trusted server postpones the servicing of the user request for a small period of time.

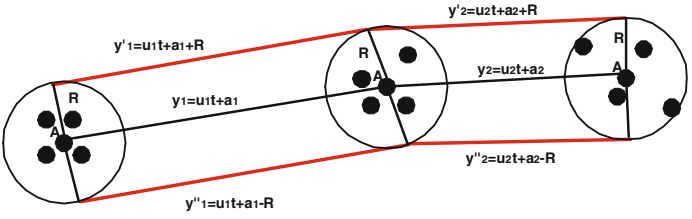

Fig. 2. 2-D Buffer and Boundary Trajectories y' and y'' of mobile user A

So, the RQ service depicted in figure 1, was transformed to the Privacy-Aware RQ service depicted in the figure 3:

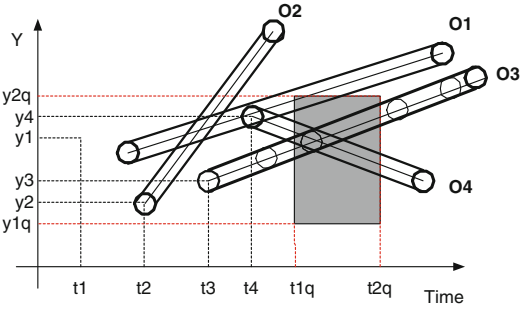

Fig. 3. Boundary Trajectories and query in (t, y) plane

If the whole buffer lies inside the query area (see the buffers of mobile users O_3 or O_4 in figure 3), meaning that both upper and lower boundary poly-lines lie in the query rectangle then the same holds for the original trajectory. If the whole buffer lies outside the query area (see the buffer of mobile user O_2 in figure 3), meaning that both upper and lower boundary poly-lines lie outside the query rectangle then the same holds for the original trajectory. In the worst-case, we face the distortion effect, where one of the two boundary poly-lines lie in the query rectangle (see the buffer of mobile user O_1 in figure 3). In the later case, we have to check what happens with the original trajectory.

4 Duality Transformation of Boundary-Trajectories and Information Distortion

The duality transform, in general, maps a hyper-plane h from R^n to a point in R^n and vice-versa. In this subsection we briefly describe how we can address the

problem at hand in a more intuitive way, by using the duality transform on the 1-d case.

4.1 Hough-X Transform

One duality transform for mapping the line with equation $y(t) = ut + a$ to a point in R^2 is by using the dual plane, where one axis represents the slope u of an objects trajectory (i.e. velocity), whereas the other axis represents its intercept a. Thus we get the dual point (u, a) (this is the so called *Hough-X transform* [11,13]). Accordingly, the 1-d query $[(t_{1_q}, t_{2_q}), (y_{1_q}, y_{2_q})]$ becomes a polygon in the dual space. By using a linear constraint query, the initial query $[(t_{1_q}, t_{2_q}), (y_{1_q}, y_{2_q})]$ in the (t, y) plane is transformed to the following rectangular query $[(u_{min}, u_{max}), (y_{1_q} - t_{1_q} u_{max}, y_{2_q} - t_{2_q} u_{min})]$ in the (u, a) plane. In a similar way, for the upper $(y'(t) = ut + a + R)$ and lower $(y''(t) = ut + a - R)$ boundary trajectories, we get the dual points $(u, a + R)$ and $(u, a - R)$ as well as the final (transformed) rectangular queries become $[(u_{min}, u_{max}), (y_{1_q} - R - t_{1_q} u_{max}, y_{2_q} - R - t_{2_q} u_{min})]$ and $[(u_{min}, u_{max}), (y_{1_q} + R - t_{1_q} u_{max}, y_{2_q} + R - t_{2_q} u_{min})]$ respectively in the (u, a) plane.

4.2 Hough-Y Transform

By rewriting the equation $y = ut + a$ as $t = \frac{1}{u} y - \frac{a}{u}$, we can arrive to a different dual representation (the so called *Hough-Y transform* in [11,13]). The point in the dual plane has coordinates (b, n), where $b = -\frac{a}{u}$ and $n = \frac{1}{u}$. Coordinate b is the point where the line intersects the line $y = 0$ in the primal space. By using this transform horizontal lines cannot be represented. Similarly, the Hough-X transform cannot represent vertical lines. Nevertheless, since in our setting lines have a minimum and maximum slope (velocity is bounded by $[u_{min}, u_{max}]$), both transforms are valid. Similarly, the initial query $[(t_{1_q}, t_{2_q}), (y_{1_q}, y_{2_q})]$ in the (t, y) plane can be transformed to the following rectangular query in the (b, n) plane: $[(t_{1_q} - \frac{y_{2_q}}{u_{min}}, t_{2_q} - \frac{y_{1_q}}{u_{max}}), (\frac{1}{u_{max}}, \frac{1}{u_{min}})]$. In a similar way for the upper $(y'(t) = ut + a + R)$ and lower $(y''(t) = ut + a - R)$ boundary trajectories, we get the transformed rectangular queries $[(t_{1_q} - \frac{y_{2_q} - R}{u_{min}}, t_{2_q} - \frac{y_{1_q} - R}{u_{max}}), (\frac{1}{u_{max}}, \frac{1}{u_{min}})]$ and $[(t_{1_q} - \frac{y_{2_q} + R}{u_{min}}, t_{2_q} - \frac{y_{1_q} + R}{u_{max}}), (\frac{1}{u_{max}}, \frac{1}{u_{min}})]$ respectively in the (b, n) plane.

4.3 The Proposed Algorithm for Privacy-Aware Indexing

Let $S = \{y_1, y_2, \ldots, y_n\}$ be the initial set of original trajectory equations, and $S' = \{y'_1, y''_1, y'_2, y''_2 \ldots, y'_n, y''_n\}$ the set of boundary trajectory equations defined by the buffer.

Algorithm 1. Index-Building

1: Decompose the 2-d motion into two 1-d motions on the (t, x) and (t, y) planes;
2: For each projection, build the corresponding index structure;

Then, let $H_xS = \{(u_1, a_1 + R), (u_1, a_1 - R), \ldots, (u_n, a_i + R), (u_n, a_i - R)\}$ and $H_yS = \{b'_1, b''_1, \ldots, b'_n, b''_n\}$ be the set of dual Hough-X and Hough-Y transforms respectively.

Algorithm1 depicts the procedure for building the index, Algorithm2 presents the procedure for Partitioning the Mobile Users according to their velocity, and Algorithm3 outlines the privacy-aware algorithm for answering the exact 2-d RQ query:

Algorithm 2. Mobile-User-Partitioning

1: Users with small velocity are stored using the Hough-X dual transform;
2: The rest are stored using the Hough-Y dual transform;
3: Motion information about the other projection is also included;

Algorithm 3. Privacy-Aware-RQ Query

1: Decompose the query into two 1-d queries, for the (t, x) and (t, y) projection;
2: For each projection get the dual-simplex query;
3: For each projection calculate a specific criterion c (for details see [11,13]) and choose the one (say p) that minimizes it;
4: For all dual-points of H_xS or H_yS sets, search in projection p the simplex query of the Hough-X or the MBR of the simplex query of the Hough-Y partition. In the latter case, perform a refinement or filtering step "on the fly", by using the whole motion information;
5: if the dual-points of both upper and lower boundary trajectories $((u_i, a_i + R), (u_i, a_i - R)$ or $b'_i, b''_i)$ lie inside the dual-simplex spatial area then the same holds for the dual-point $((u_i, a_i)$ or $b_i)$ of the original trajectory;
6: else if the dual-points of both upper and lower boundary trajectories lie outside the dual-simplex spatial area then the same holds for the dual-point of the original trajectory;
7: else having in mind the value R, search the simplex query of the Hough-X or Hough-Y partition for the dual-points of original trajectories;

In [11,13], $Q_{Hough-X}$ is computed by querying a 2-d partition tree, whereas $Q_{Hough-Y}$ is computed by querying a B^+-tree that indexes the b parameters. Here, we consider the case, where the users are moving with non small velocities u, meaning that the velocity value distribution is skewed (Zipf) towards u_{min} in some range $[u_{min}, u_{max}]$ and as a consequence the $Q_{Hough-Y}$ transformation is used (denote that u_{min} is a positive lower threshold). Moreover, our method will incorporate the Lazy B-tree [10] indexing scheme, since the latter can handle update queries in optimal (constant) number of block-transfers (I/Os). As a result, we get Algorithm 4.

Let K be the number of b_i parameters associated to boundary trajectories of buffers that intersect with the query rectangle. Then, algorithm 4 requires $T(n) = O(Cost(LazyB\_tree) + K)$ I/Os or block transfers. Moreover, and according to notations presented in [1], let say D be the initial Database that

Algorithm 4. Privacy-Aware Indexing of $Q_{Hough-Y}$ partition with Lazy B-tree

1: BEGIN_PSEUDOCODE
2: Decompose the query into two 1-d queries, for the (t, x) and (t, y) projection;
3: For each projection get the dual-simplex query;
4: Search the MBR of the simplex query of the Hough-Y partition and perform a filtering step "on the fly", by using the whole motion information;
5: Let $B \subset H_y S$ be the answer set of dual parameters, which satisfy the query above;
6: $Result = 0$;
7: For all elements of B do
8: Begin_for
9: if ($b_i' \in B$ AND $b_i'' \in B$) then
$Result = Result \cup (idofuser_i, neighbours(idofuser_i, k-1)$;
10: return Result;
11: else if ($b_i' \notin B$ AND $b_i'' \notin B$) then return Result;
12: else begin
13: if $b_i \in MBR$ of Hough-Y simplex partition then
14: $Result = Result \cup (idofuser_i, neighbours(idofuser_i, k-1)$;return Result;
15: else return Result;
16: end
17: End_for
18: END_PSEUDOCODE

stores the N original trajectories and D' the Privacy-Aware Database that stores the $2N$ boundary-trajectories. Let also say, $Q(D)$ and $Q(D')$ be the Query Results obtained consuming $T(D)$ and $T(D')$ block-transfers (I/Os) in D and D' database schemes respectively. We define as $Distortion\_Ratio = \frac{|Q(D)-Q(D')|}{max(Q(D),Q(D'))}$ and as $Competitive\_Ratio = \frac{|T(D)-T(D')|}{max(T(D),T(D'))}$. In the most of the cases, $T(D') > \ldots > T(D)$, thus it is very important to find out, how competitive to the optimal one ($T(D)$) is the privacy-aware method that answers the query in D'. Since, the distortion effect in D' absolutely depends on parameter K, an experimental evaluation of ***Competitive_Ratio vs*** K is also presented in the following section.

5 Experimental Evaluation

This section compares the query performance of our privacy-aware method, when incorporates STRIPES [14] (the best known solution), Lazy B-trees (LBTs) and TPR*-tree, respectively. We deploy spatio-temporal data that contain insertions at a single timestamp 0. In particular, objects' MBRs are taken from the LA spatial dataset (128971 MBRs)[1]. We want to simulate a situation where all objects move in a space with dimensions 100x100 kilometers. For this purpose each axis of the space is normalized to [0,100000]. For the TPR*-tree, each object is associated with a VBR (Velocity Bounded Rectangle) such that (a) the object

[1] Downloaded from the Tiger website `http://www.census.gov/geo/www/tiger/`

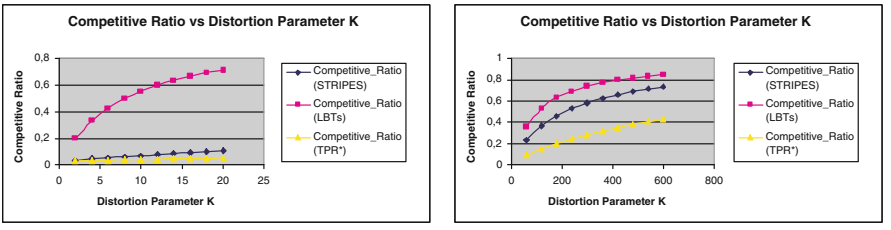

Fig. 4. $q_V len = 5$, $q_T len = 50$ $q_R len = 100$ (top), $q_R len = 1000$ (bottom), $R_{max} = 50$ (top), $R_{max} = 200$ (bottom)

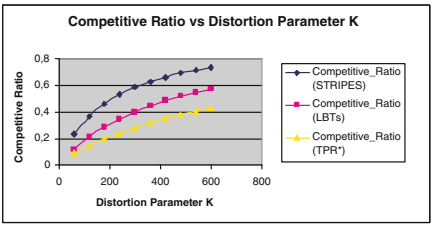

Fig. 5. $q_R len = 2000$, $q_V len = 5$, $q_T len = 50$, $R_{max} = 500$

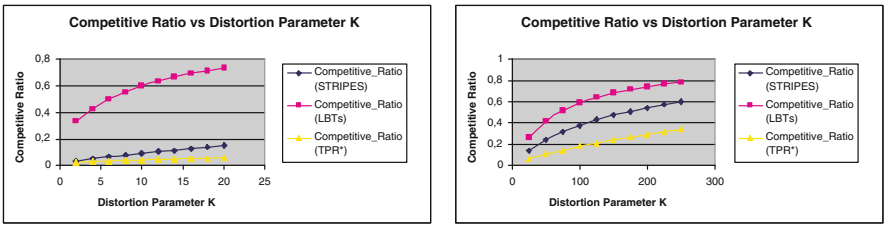

Fig. 6. $q_V len = 10$, $q_T len = 50$, $q_R len = 400$ (top), $q_R len = 1000$ (bottom), $R_{max} = 100$ (top), $R_{max} = 200$ (bottom)

does not change spatial extents during its movement, (b) the velocity value distribution is skewed (Zipf) towards 30 in range [30,50], and (c) the velocity can be either positive or negative with equal probability. As in [2], we will use a small page size so that the number of index nodes simulates realistic situations.

Thus, for all experiments, the page size is 1 Kbyte, the key length is 8 bytes, whereas the pointer length is 4 bytes. Thus, the maximum number of entries ($< x >$ or $< y >$, respectively) in both Lazy B-trees and B$^+$-trees is 1024/(8+4)=85. In the same way, the maximum number of entries (2-d rectangles or $< x1, y1, x2, y2 >$ tuples) in TPR*-tree is 1024/(4*8+4)=27. On the other hand, the STRIPES index maps predicted positions to points in a dual transformed space and indexes this space using a disjoint regular partitioning

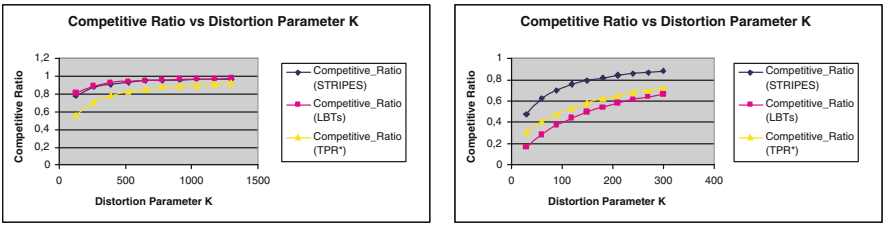

Fig. 7. $q_V len = 5$, $q_T len = 1$, $q_R len = 400$ (top), $q_R len = 1000$ (bottom), $R_{max} = 100$ (top), $R_{max} = 200$ (bottom)

of space. Each of the two dual planes, are equally partitioned into four quads. This partitioning results in a total of $4^2 = 16$ partitions, which we call *grids*. The fanout of each non-leaf node is thus 16. For each dataset, all indexes except for STRIPES have similar sizes. Specifically, for LA, each tree has 4 levels and around 6700 leaves apart from STRIPES index which has a maximum height of seven and consumes about 2.4 times larger disk space. Each query q has three parameters: $q_R len$, $q_V len$, and $q_T len$, such that (a) its MBR q_R is a square, with length $q_R len$, uniformly generated in the data space, (b) its VBR is $q_V = -q_V len/2, q_V len/2, -q_V len/2, q_V len/2$, and (c) its query interval is $q_T = [0, q_T len]$. The query cost is measured as the average number of node accesses in executing a workload of 200 queries with the same parameters. Implementations were carried out in C++ including particular libraries from SECONDARY LEDA v4.1.

5.1 Query Cost Comparison

We measure the Competitive Ratio of LBTs method (this method incorporates two Lazy B-trees that index the appropriate b parameters in each projection respectively, and finally combines the two answers by detecting and filtering all the pair permutations), the TPR*-tree and STRIPES presented in [18] and [14] respectively, using the same query workload, after every 10000 updates. Figures 4 up to 8 show the Competitive Ratio as a function of K (for datasets generated from LA as described above), using workloads with different parameters. Parameter K represents boundary trajectories of buffers that intersect with the query rectangle, and obviously require an additional filtering on the fly process. Obviously, the required number of block transfers depends on the answer's size as well as the size of K.

Figure 4 depicts how competitive to the optimal solution the LBTs method is, in comparison to TPR*-tree and STRIPES. The Ratio degrades as the query rectangle length grows from 100 to 1000. When the query rectangle length or equivalently the query surface becomes extremely large (e.g. 2000), then the STRIPES index becomes more competitive (see Figure 5).

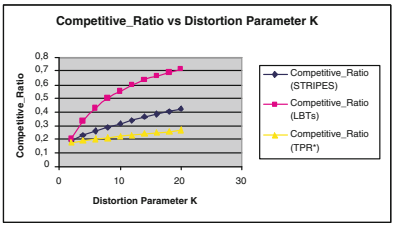

Fig. 8. $q_R len = 400$, $q_V len = 5$, $q_T len = 100$, $R_{max} = 100$

Figure 6 depicts how competitive to the optimal solution the LBTs method is, towards to TPR*-tree and STRIPES, in case the velocity vector grows. The Ratio degrades as the query rectangle length grows from 400 to 1000.

Figure 7 depicts the performance of all methods in case the time interval length degrades to value 1. Even in this case, the LBTs method is more competitive than STRIPES and TPR*-tree. As query rectangle length grows from 400 to 1000, the LBTs method advantage decreases; we remark that STRIPES becomes faster, whereas LBTs method has exactly the same performance with the TPR*-trees.

Figure 8 depicts the efficiency of LBTs solution in comparison to that of TPR*-trees and STRIPES respectively in case the time interval length enlarges to 100.

6 Conclusions

We presented the problem of anonymity preserving data publishing in moving objects databases. In particular, we studied the case where the trajectory of a mobile user on the plane is no longer a polyline in a two-dimensional space, instead it is a two-dimensional surface. By transforming the surface's boundary poly-lines to dual points we experimentally focused on the impact of information distortion introduced by this space translation.

References

1. Abul, O., Bonchi, F., Nanni, M.: Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. In: Proceedings 24th IEEE International Conference on Data Engineering, Cancun, pp. 376–385 (2008)
2. Beckmann, N., Kriegel, H.P., Schneider, R., Seeger, B.: The R*-tree: an Efficient and Robust Access Method for Points and Rectangles. In: Proceedings ACM International Conference on Management of Data (SIGMOD), Atlantic City, NJ, pp. 322–331 (1990)
3. Bajaj, R., Ranaweera, S., Agrawal, D.: GPS: location-tracking technology. Computer 35(4), 92–94 (2002)
4. Clarke, R.: Person location person tracking - technologies risks and policy implications. Information Technology and People 14(2), 206–231 (2001)

5. D'Roza, T., Bilchev, G.: An overview of location-based services. BT Technology Journal 21(1), 20–27 (2003)
6. Gkoulalas-Divanis, A., Verykios, V.S., Bozanis, P.: A network aware privacy model for online requests in trajectory data. Data Knowl. Eng. 68(4), 431–452 (2009)
7. Gaede, V., Gunther, O.: Multidimensional Access Methods. ACM Computing Surveys 30(2), 170–231 (1998)
8. Hoh, B., Gruteser, M., Xiong, H., Alrabady A.: Preserving privacy in gps traces via uncertainty-aware path cloaking. In: ACM Conference on Computer and Communications Security, pp. 161–171 (2007)
9. Jensen Christian, S., Lin, D., Ooi, B.C.: Query and Update Efficient B+-Tree Based Indexing of Moving Objects. In: VLDB 2004, pp. 768–779 (2004)
10. Kaporis, A., Makris, C., Sioutas, S., Tsakalidis, A., Tsichlas, K., Zaroliagis, K.: ISB-Tree: a New Indexing Scheme with Efficient Expected Behaviour. In: Deng, X., Du, D.-Z. (eds.) ISAAC 2005. LNCS, vol. 3827, pp. 318–327. Springer, Heidelberg (2005)
11. Kollios, G., Gunopulos, D., Tsotras, V.: On Indexing Mobile Objects. In: Proceedings 18th ACM Symposium on Principles of Database Systems (PODS), Philadelphia, PA, pp. 261–272 (1999)
12. Manolopoulos, Y., Theodoridis, Y., Tsotras, V.: Advanced Database Indexing. Kluwer Academic Publishers, Dordrecht (2000)
13. Papadopoulos, D., Kollios, G., Gunopulos, D., Tsotras, V.J.: Indexing Mobile Objects on the Plane. In: Hameurlain, A., Cicchetti, R., Traunmüller, R. (eds.) DEXA 2002. LNCS, vol. 2453, pp. 693–697. Springer, Heidelberg (2002)
14. Patel, J., Chen, Y., Chakka, V.: STRIPES: an Efficient Index for Predicted Trajectories. In: Proceedings ACM International Conference on Management of Data (SIGMOD), Paris, France, pp. 637–646 (2004)
15. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Transactions on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
16. Sweeney, L.: K-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10(5), 557–570 (2002)
17. Salzberg, B., Tsotras, V.J.: A Comparison of Access Methods for Time-Evolving Data. ACM Computing Surveys 31(2), 158–221 (1999)
18. Tao, Y., Papadias, D., Sun, J.: The TPR*-Tree: an Optimized Spatio-Temporal Access Method for Predictive Queries. In: Proceedings 29th International Conference on Very Large Data Bases (VLDB), Berlin, Germany, pp. 790–801 (2003)