

Supporting Pattern Matching Queries over Trajectories on Road Networks

Gook-Pil Roh, Jong-Won Roh, Seung-Won Hwang, and Byoung-Kee Yi

Abstract—With the advent of ubiquitous computing, we can easily collect large scale trajectory data, say, from moving vehicles. This paper studies pattern matching problems for trajectory data over road networks, which complements existing efforts focusing on (a) a spatio-temporal window query for location-based service or (b) Euclidean space with no restriction. In contrast, we first identify some desirable properties for pattern matching queries to the road network trajectories. As the existing work does not fully satisfy these properties, we develop (1) trajectory representation and (2) distance metric that satisfy all the desirable properties we identified. Based on this representation and metric, we develop efficient algorithms for three types of pattern matching queries— whole, subpattern, and reverse subpattern matching. We analytically validate the correctness of our algorithms and also empirically validate their scalability over large-scale, real-life and synthetic trajectory datasets.

Index Terms—Trajectory, road network, pattern matching query

1 INTRODUCTION

WITH the advent of ubiquitous computing, we can easily acquire the locations of moving objects, *e.g.*, vehicle locations from global positioning system (GPS). Querying over the paths of such objects, or *trajectories*, has thus gained attention lately. However, most of the previous research efforts focus on efficiently evaluating the *spatio-temporal query*, such as supporting range and K nearest neighbor (KNN) queries, from the given query point, for location-based services (*e.g.*, [1]). In [2], many index structures are surveyed for efficient query processing on the spatio-temporal database.

In addition, prior work typically assumes objects can move anywhere and considers Euclidean space as search space. However, in many real-life applications, the movements are often constrained by obstacles such as buildings or trees. To reflect such constraints in query processing, recent work [3], [4], [5] studied moving objects with constraints on movement.

In this paper, we aim at supporting effective queries over trajectories, overcoming these two limitations of prior research identified above. *First*, we support new types of queries that search for matching trajectory patterns in a database. While such pattern matching queries have been successfully adopted for querying time series data, our work, to the best of our knowledge, is the first to support pattern matching queries over trajectory data. *Second*, we focus on moving objects with constraints

on their movements. In particular, we deal with the trajectories of vehicles or mobile users on the road. Such moving objects can move only along the roads, thus their trajectories are constrained by the *road network*. Such a trajectory is known as the *network trajectory*.

Toward the goal, we first identify five desirable properties that have not been fully satisfied by any of the prior work:

R1: Road network. As we assume that the moving object moves only along the road, the moving object should not be located off the road, (see Fig. 1a), and such off-road locations should not affect measuring the distance between trajectories.

R2: Spatial proximity. The distance measure between trajectories should reflect the spatial proximity from the viewpoint of the road network. For instance, in Fig 1b, the proximity between *B* and *C* shares more road segments than *A* and *C*, which has to be reflected to the distance measure.

R3: Sampling rate / speed invariant. Due to the difference in sampling rates or speeds, two objects moving along the same route could be represented by two different trajectories (see *A* and *B* in Fig. 1c). Even when the two objects have the same sampling rates and move along the same route, if the sampling is not synchronized, their trajectories can still differ (see *C* and *D* in Fig. 1c). The distance measure should not be affected by when or how often the locations are sampled.

R4: Robust to noise. Due to measurement errors or communication failures, trajectories may contain noises, as illustrated in Fig. 1d. If a distance measure is sensitive to noise, it cannot reflect the similarity between *B* and *C*, and may report *A* is closer to *C*. To avoid the problem, the distance measure should be robust to noise, in order to identify an unusual movement as noise and eliminate it in similarity computation.

R5: Metric. In general, it is desirable for distance mea-

-
- G. Roh and J. Roh are with the Department of Computer Science & Engineering, POSTECH, Pohang, Korea.
E-mail: {noh9pil,nbanoh}@postech.ac.kr
 - Corresponding author. S. Hwang is with the Department of Computer Science & Engineering, POSTECH, Pohang, Korea.
E-mail: {swhwang}@postech.ac.kr
 - Corresponding author. B. Yi is with IHIS Research Center, Kyungpook National University, Daegu, Korea.
E-mail: byoungkeeyi@gmail.com

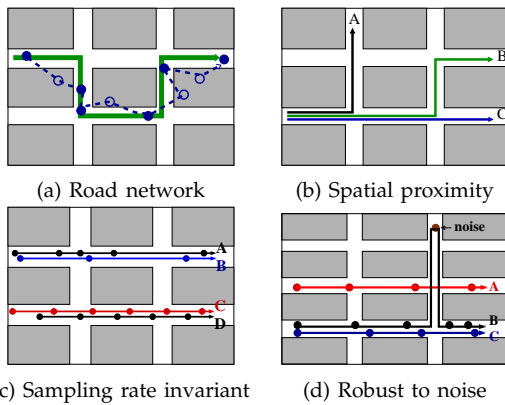


Fig. 1. Illustration of desirable properties identified

tures to be *metric*, because irrelevant objects can be pruned out with no false dismissal leveraging existing index structures. However, this property is not required but rather recommended, as non-metric measures can achieve the same advantages, if appropriate indexing schemes can be developed [6].

Based on the above requirements, we propose a desirable trajectory representation and distance metric that satisfy all requirements. In particular, we consider three types of matching queries, which we illustrate with examples below.

Whole pattern matching: Whole pattern matching captures the global similarity between two trajectories and compares them in their entireties without ignoring any part of them. Consider two example trajectories in Fig. 2. Suppose trajectory T_1 represents a commute from work to home of some user U_1 . He can then query with T_1 to retrieve other trajectories sharing similar road segments, e.g., T_2 to carpool with.

Subpattern matching: Subpattern matching on the other hand considers the query trajectory as whole and some interesting parts of database trajectories that best match the query trajectory and ignores the rest of them. To illustrate, suppose there is a road construction plan, say S_3 in Fig. 2. Then, the drivers who may be affected by the construction are those whose commute routes contain S_3 as a subpattern, which is T_2 in this particular case.

Reverse subpattern matching: Alternatively, the subpattern matching problem can be reversed. Suppose we consider a plan for a new bus route. Reverse subpattern matching can be useful to find the drivers who may use the bus instead of driving to commute. For example, if T_1 in Fig. 2 is the new bus route, then S_1 and S_2 are the candidates.

In summary, our contributions are as follows: (a) pattern matching over trajectories in road networks; (b) identification of desirable properties; (c) a trajectory representation and similarity measure that satisfy the properties; (d) efficient algorithms for whole, subpattern, and reverse subpattern matchings; and (e) validation of the quality and efficiency of the proposed methods using both synthetic and real datasets.

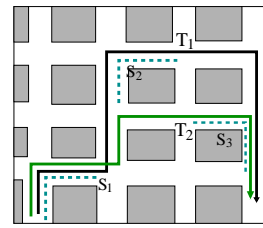


Fig. 2. Illustration of three pattern matching types

The rest of the paper is organized as follows. Section 2 presents the prior research work. We propose trajectory representation in Section 3 and distance measure in Section 4. Section 5 describes our matching algorithms. Section 6 reports our experimental results. Section 7 concludes the paper.

2 RELATED WORK AND BACKGROUND

In this section, we survey prior research efforts on (a) querying trajectories and (b) modeling trajectories and their similarity. We also present some background information necessary for the proposed methods.

Queries: There have been prior research efforts for querying trajectories constrained by networks [3], [4], [5], [7]. A data model of network-constrained moving objects has been proposed in [8], followed by many index structures proposed to efficiently perform the spatio-temporal window query (3D interval) in [3], [4], [5]. Lin *et al.* [9] and Frenzos *et al.* [10] proposed a similarity search of moving object trajectories, which is similar to our whole pattern matching query. In a clear contrast, this paper focuses on whole, subpattern, and reverse subpattern matching queries, of which the last two types of queries have never been studied before for trajectory data.

Trajectory representation and similarity: Next, we survey prior work on the trajectory and similarity modeling, which we categorize into two categories—the sequence based [11], [12], [13], [14], [15], [16], [17] and the shape-based similarity measures [9], [18], [19], [20]. The sequence-based similarity measures build upon trajectories represented as a *sequence* of points in two or three dimensional space, comparing how closely the two sequences align with each other. In contrast, shape-based similarity measures build upon trajectories as a *set* of points or line segments and they are thus not sensitive to the order of points or segments. We stress that, while these measures do not consider the road network, our work, for the first time to the best of our knowledge, supports pattern matching queries for the network-constrained trajectories.

We summarize the fulfillment of requirements for the existing distance measures in Table 1. The symbols ‘O’, ‘X’, and Δ mean ‘satisfy’, ‘not satisfy’, and ‘partially satisfy’ respectively.

TABLE 1
Summary of previous research

| Requirements | | R1 | R2 | R3 | R4 | R5 |
|--------------|------------------------------------|----|----------|----------|----|----|
| Sequence | LCSS [12],EDR [13] | X | Δ | X | O | X |
| | ERP [14] | X | O | X | X | O |
| | FLOCK [15],CONVOY [16] | X | X | X | X | X |
| | TS2 [17] | X | Δ | Δ | X | O |
| Shape | Hausdorff (Lee <i>et al.</i> [18]) | X | Δ | X | X | O |
| | Frèchet [19] | X | O | X | X | O |
| | normalization [20] | X | Δ | O | X | X |
| | Lin <i>et al.</i> [9] | X | X | X | O | X |

3 TRAJECTORY REPRESENTATION

In this paper, we propose to represent a trajectory as a sequence of road segments in a road network. We define a road segment as its starting and ending positions and require it not to intersect with any other segment. (Note that this requirement is not restrictive, since a crossing segment pair can be alternatively represented as four non-intersecting segments.) We also require a trajectory to be *connected* in that the ending point of a segment should be the starting point of the next segment in the sequence. With this representation, we can represent the restriction of road networks and the spatial proximity of road segments, without being affected by different sampling rates of moving objects.

Trajectories of *raw data* from positioning devices such as GPS consist of quadruplets (x, y, p, v) , where x and y are 2D location points, while p and v are the time stamp and the velocity respectively. Most of the prior researches take the raw data “as is” as input, which cannot represent the restrictions of the underlying road networks.

In contrast, our proposed trajectory representation uses a sequence of road segments, which requires a pre-processing phase to transform the raw data into a connected sequence of road segments. One alternative is to adopt map-matching algorithms for this process, especially an *incremental* map-matching algorithm [21] devised for fast map-matching. However, this algorithm requires the connectivity information, such as an adjacency matrix, of the entire road networks, let alone the need to tune five parameters used for scoring. In contrast, we devise a memory-efficient scheme, requiring to maintain only the vicinity area (which is less than 0.01% of the entire network in our evaluations).

To devise an efficient mapping, there are two challenges: *First*, for each 2D point, we need to find the road segment it belongs to. *Second*, in case two consecutive road segments are not connected, which can happen due to the error of the GPS device or infrequent sampling, we need to connect the two, to satisfy our assumption.

For the first challenge, we adopt R-tree [22] to index all the road segments of the given road network. This approach is similar to the method that employs the R-tree index to find the road segment that covers a query point in [23]. Each road segment is transformed to the Minimum Bounding Rectangle (MBR) whose diagonal

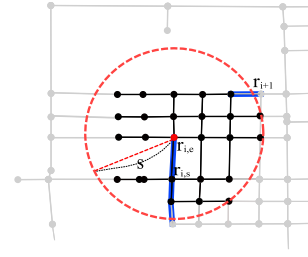


Fig. 3. Example of vicinity circle/graph

is the straight line, connecting $r_{i,s}$ and $r_{i,e}$ of road segment, and sides are parallel to the coordinate axes. Once these MBRs are inserted into the R-tree, the nearest road segment for the given 2D point can be identified by simply performing a nearest neighbor search on the R-tree.

For the second challenge of connecting two *disconnected* consecutive road segments, when the ending point of one segment does not match the starting point of the next, we use the shortest path approach that recovers the missing path by finding the shortest path between the ending position and the starting position of the two segments. This approach is based on an assumption that a moving object insists on taking the shortest path to reach its destination. For this solution, we can employ one of the heuristic algorithms listed in [24], adopting existing shortest path algorithms on large-scale real road networks.

However, running a shortest path algorithm, such as Dijkstra’s, for the entire road network incurs prohibitive cost, *i.e.*, $O(n^2)$, where n is the number of nodes in the graph representing the road network. Considering there are usually millions of nodes in the road network, we need to restrict the search space. More specifically, we restrict our search to road segments in the “vicinity circle” of the disconnected road segments. To ensure correctness, we generate a vicinity graph G that includes all road segments on the shortest path.

To generate a vicinity graph G , for the given disconnected road segments r_i and r_{i+1} , we find a vicinity circle, centered at the ending position of the preceding road segment, *i.e.* $r_{i,e}$. Fig. 3 illustrates the vicinity circle of the disconnected road segment r_i and r_{i+1} with its radius $s = v_{max} \times (ts(r_{i+1}) - ts(r_i))$ where v_{max} is the maximum speed of the moving object and $ts(r_i)$ is the time stamp of sample point which is transformed to r_i . If r_i includes multiple sample points, we take the last one. Note that, this circular region covers the maximum *driving* distance a moving object can travel along the roads from r_i to r_{i+1} . All the road segments contained by the vicinity circle are retrieved (*e.g.*, the road segments in dark in Fig. 3) and then inserted into the vicinity graph G .

Once the vicinity graph is identified, we run a shortest path algorithm to find the shortest path segments from $r_{i,e}$ to r_{i+1} . In the case r_{i+1} is disconnected from r_{i+2} , it is not clear which endpoint is the starting point. In

such a case, we should identify the shortest path from the single source point to both endpoints of r_{i+1} , i.e., “one-to-some” shortest path. For such a case, we choose the Dijkstra’s approximate bucket implementation [25], which was empirically validated in [26] to perform well for one-to-some shortest path problem.

To eliminate noise caused by measurement errors, we reasonably assume that noise does not occur in k consecutive road segments, possibly with the help of error correction technology of GPS devices. (k is an application dependent parameter.) With this assumption, we calculate the vicinity area until we find the road segment which is intersected by the vicinity area in k consecutive road segments from r_i . If r_{i+k} is the road segment that is first intersected by the vicinity area, we drop the road segments from r_{i+1} to r_{i+k-1} .

4 DISTANCE MEASURE

In this section, we present a novel distance measure between two trajectories. In particular, we extend the intuition of Hausdorff distance to our trajectory representation– Informally, the Hausdorff distance between two sequences of segments, is the longest distance an adversary can force you to travel from one segment to another. More specifically, we first determine how to measure distance between two road segments. Using this measure, we then define the distance measure for trajectories.

First, we define the distance $d(r_i, r_j)$ between two road segments r_i and r_j , based on which we will define the distance between the trajectories later on. The distance between road segments is defined as follows:

Definition 1 (Road segment distance):

$$d(r_i, r_j) = \max \left\{ \vec{d}(r_i, r_j), \vec{d}(r_j, r_i) \right\}, \quad (1)$$

where $\vec{d}(r_i, r_j)$ is a *one-way* road segment distance from r_i to r_j which is defined below.

Definition 2 (one-way road segment distance):

$$\vec{d}(r_i, r_j) = \max \left\{ \begin{array}{l} \min \{ \|r_{i,s}, r_{j,s}\|, \|r_{i,s}, r_{j,e}\| \}, \\ \min \{ \|r_{i,e}, r_{j,s}\|, \|r_{i,e}, r_{j,e}\| \} \end{array} \right\}, \quad (2)$$

where $\|a, b\|$ is the Euclidean distance between a and b . The road segment distance is the longest of two possible one-way distances to travel from one segment to the other.

One may argue that $\|a, b\|$ should be the shortest path distance between a and b , to satisfy the **R1** property. Such an alternative, however, incurs prohibitive computational cost, because we have to search a large portion of the road network if a and b are road segments located far from each other. Therefore, the shortest path distance is practically infeasible for comparing trajectories.

Based on the road segment distance above, we now move on to define the distance measure between two trajectories.

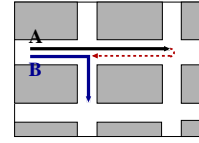


Fig. 4. Meaning of proposed distance measure

Definition 3 (Trajectory distance): Given two trajectories $T_a = [a_1, \dots, a_n]$ and $T_b = [b_1, \dots, b_m]$, the distance between them is defined as follows:

$$D(T_a, T_b) = \max \left\{ \vec{D}(T_a, T_b), \vec{D}(T_b, T_a) \right\}, \quad (3)$$

where $\vec{D}(T_a, T_b)$ is *one-way* trajectory distance from T_a to T_b .

Definition 4 (One-way trajectory distance): Given two trajectories $T_a = [a_1, \dots, a_n]$ and $T_b = [b_1, \dots, b_m]$, the *one-way* trajectory distance is defined by

$$\vec{D}(T_a, T_b) = \max_{a_i \in T_a} \min_{b_j \in T_b} d(a_i, b_j) \quad (4)$$

Similar to Hausdorff distance, our distance measure can be explained as the longest path from each road segment to its closest road segment in another trajectory. To illustrate, consider two trajectories of moving objects A and B in Fig. 4. The two trajectories share the same movement up to the point where A keeps moving straight while B makes a right turn. That is, the distance between A and B is 0, until the two objects take different paths, when the distance to travel for A to reach B , or vice versa, is either the distance of A making a U-turn to B , as represented as a dashed line in the figure, or B making a U-turn to A . As the former distance is longer, we use the length of the dashed line as the distance between two trajectories.

Next, we argue how our proposed representation and distance metric fulfill the properties identified in Section 1. First, our trajectory representation as a sequence of connected road segments, satisfies the three requirements **R1**, **R2**, and **R3**– Our representation considers the restriction of road networks and the spatial proximity of the segments, and it is not affected by the different sampling rates of the objects. Second, we argue how our proposed distance measure also satisfies **R4** and **R5**.

While our distance itself does directly satisfy **R4** property, we achieve it by means of the robust trajectory representation as described in the previous section. Lastly, we show that our distance measure is a metric, which ensures efficient query processing using an index structure, as we will later discuss.

Theorem 1: The trajectory distance is a metric.

Proof: The reflexivity and symmetry directly follow from Eq. 3 and 4. To prove the triangle inequality, we first define ϵ -neighborhood road segments of trajectory T_a , denoted by $N_\epsilon(T_a)$, as follows: $N_\epsilon(T_a) = \{r \in R | d(r, a_i) \leq \epsilon, 1 \leq i \leq n\}$. Then,

$$\begin{aligned}
\vec{D}(T_a, T_c) &= \inf \{ \epsilon > 0 | T_a \subset N_\epsilon(T_c) \} \\
&= \inf \{ \delta + \delta' | T_a \subset N_{\delta+\delta'}(T_c) \} \\
&\leq \inf \{ \delta + \delta' | T_a \subset N_\delta(T_b) \text{ and } T_b \subset N_{\delta'}(T_c) \} \\
&\leq \inf \{ \delta | T_a \subset N_\delta(T_b) \} + \inf \{ \delta' | T_b \subset N_{\delta'}(T_c) \} \\
&= \vec{D}(T_a, T_b) + \vec{D}(T_b, T_c).
\end{aligned}$$

Similarly, $\vec{D}(T_c, T_a) \leq \vec{D}(T_b, T_a) + \vec{D}(T_c, T_b)$. \square

5 PATTERN MATCHING QUERIES

Given a query trajectory, we support three types of pattern matching: whole, subpattern, and reverse subpattern matching, for finding “similar” trajectories to a query trajectory in the database. However, as illustrated in Section 1, for each type of matching, the definition of “similar” trajectory varies. First, in whole pattern matching, the query trajectory is typically of similar length to the trajectories in the database and the query retrieves the result trajectories that are globally similar to the query trajectory. Second, in subpattern matching, the query trajectory is typically much shorter than the trajectories in the database and the query retrieves the results containing some subpatterns that are similar to the query pattern. Lastly, in reverse subpattern matching, the query is typically much longer than the trajectories in the database and the query retrieves the trajectories in the database that match a certain part of the query pattern.

For whole pattern matching, we can simply employ the range and KNN search algorithm of M-tree [27] without any modification, as our distance measure is metric (Theorem 1). However, for subpattern matchings, we only need to consider some part of a database trajectory that matches the query the best and quantifies the (dis)similarity. Let T_q and T be the query and a database trajectory respectively. Then, the *one-way* distance, $\vec{D}(T_q, T)$, perfectly captures such a notion, which is only affected by the nearest road segment of T from each road segment of T_q . Similarly, it can also be used for reverse subpattern matching but in the opposite direction, *i.e.* $\vec{D}(T, T_q)$, by identifying the closest query segment to trajectory data.

For efficient processing of pattern matching queries, we devise pruning rules, for which we introduce some additional notations. For a nonleaf node O of M-tree, let $rp(O)$ denote its representative and $cr(O)$ be the covering radius that is the distance between $rp(O)$ and the farthest object from it in the subtree rooted by O . Then we have the following theorems, one for each type of pattern matching, that follow directly from the fact that both the one-way and two-way distance measures satisfy triangle inequality and that $cr(O)$ is the maximum (two-way) distance from O to any objects within the subtree.

Theorem 2 (Pruning condition for whole patterns): Given a query trajectory T_q and a threshold ϵ , no trajectory T in O satisfies $D(T_q, T) \leq \epsilon$, if $D(T_q, rp(O)) > cr(O) + \epsilon$.

Theorem 3 (Pruning condition for subpatterns): Given a query trajectory T_q and a threshold ϵ , no trajectory T in O satisfies $\vec{D}(T_q, T) \leq \epsilon$, if $\vec{D}(T_q, rp(O)) > cr(O) + \epsilon$.

Theorem 4 (Pruning condition for reverse subpatterns): Given a query trajectory T_q and a threshold ϵ , if $\vec{D}(rp(O), T_q) > cr(O) + \epsilon$, no trajectory T in O satisfies $\vec{D}(T, T_q) \leq \epsilon$.

6 EXPERIMENTAL RESULTS

In this section, we report the results of experiments conducted to verify the effectiveness and efficiency of our proposed framework. We begin by describing experimental settings. We used real trajectory data [28] to show the quality of the matching trajectories. There are 213 trajectories, each of which is a sequence of positions received by GPS in the Cook and Dupage county of Illinois. Additionally, small random noises were added to the original data to demonstrate the robustness of the proposed measure. We observe that there are three “ground truth” groups in the real trajectory data and the distance between inter-group members are larger than the distance between intra-group members. These trajectories were then transformed to sequences of road segments based on the method described in Section 3, with length ranging from 8 to 742. We obtained the road network data of the two counties in a TIGER/LINE format from the U.S. Census Bureau [29] consisting of 201,540 road segments in the road network. All experiments were conducted on a machine with a Pentium-4 CPU (3.2GHz) and 1 GBytes of main memory, running a version of the Linux operating system. We implemented the range and KNN search algorithms for each pattern matching query on top of the M-tree.

We now move on to discuss the result of three pattern matching queries. In the dataset, we observed that 93 trajectories share similar patterns, of which we randomly picked one as a query trajectory for both whole and reverse subpattern matching. It consists of 444 road segments, of which we also randomly extracted 49 consecutive road segments as a query trajectory for the subpattern matching.

Using the query trajectories from above, we performed all three types of pattern matchings and visualize the results in the order of similarity to the query for each type of matching. In Fig. a thick pattern indicates a query trajectory and a thin pattern indicates a matching trajectory. From the whole pattern matching results, we can observe that the top 93 results are the trajectories that either exactly match the entire query or the majority of road segments. The results ranked 81th and 87th share the majority of road segments of the query, with minor deviations in the middle. Recall that, there exist three ground-truth clusters in the real-life set. The top 93 results belong to one such cluster (travelling across counties), and the 96th and 97th results represent the other two clusters (travelling within a county), respectively. Because of this clustered nature, top 93 results are

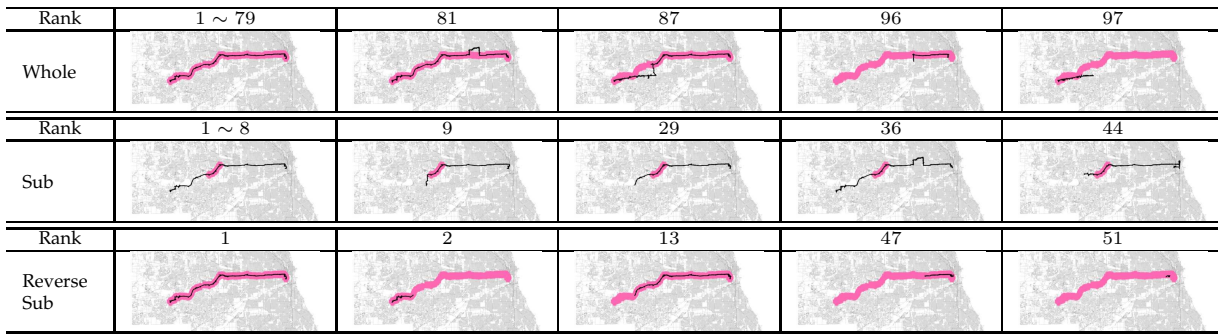


Fig. 5. Visualization of trajectories for each pattern matching

TABLE 2
Quality comparison with other measures

| Groups (# of trajectories) | Num. of Correctly Classified Trajectories (Recall %) | | | |
|----------------------------------|--|----------|----------|------------------|
| | EDR [13] | ERP [14] | OHD [9] | Proposed |
| G1 (96) | 70 (73%) | 70 (73%) | 87 (90%) | 96 (100%) |
| G2 (24) | 4 (17%) | 8 (33%) | 3 (13%) | 24 (100%) |
| G3 (93) | 49 (53%) | 60 (65%) | 67 (72%) | 93 (100%) |

highly similar to the query, while the similarity of lower ranked results deviate rather dramatically.

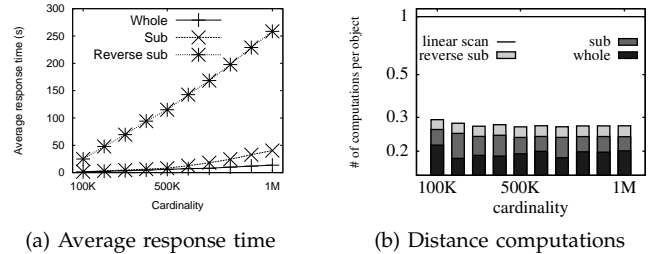
Top results from subpattern matching include longer trajectories that subsume the query pattern. Note that as this query pattern is a part of the query used for the whole pattern matching, most of them overlap with those from the whole matching. We thus highlight only the trajectories that do not overlap in Fig. 5, such as the trajectories with rank 9, 29, and 44 in the figure.

Lastly, in reverse subpattern matching, top results include the exact match and short trajectories subsumed by the query trajectory. The top-most trajectory is the query trajectory itself, as the query trajectory was selected from the trajectory database. In Fig. 5, we highlight the results that do not overlap with other query results.

Summing up, we can observe from the result trajectories of the three queries in Fig. 5 that the proposed distance metric well reflects the semantics of each pattern matching query.

We now move on to discuss the effectiveness of our distance measure comparing with other measures using real-life road network trajectories [28]. As trajectories in this set clearly fall into three groups, we use such groups as “ground truth” results and compare the recall of all metrics in Table 2. Specifically, for each group of size m , we select one trajectory in the group as query and compare the top- m results from each metric. When using our metric, we identify all trajectories in the group and nothing else (*i.e.*, perfect precision/recall), while the recall of other metric varies from 17% to 90%.

We also empirically study whether our proposed metric is in any way redundant to existing metrics, by comparing the rank correlations between measures using widely-adopted Kendall’s tau coefficient [30]. Specifically, using one trajectory from each group as a query, we rank the whole trajectories by similarity and measure



(a) Average response time

(b) Distance computations

Fig. 6. Performance of pattern matching queries

the rank similarities. If our proposed metric correlates perfectly with the existing metric, Kendall’s tau coefficient is 1, suggesting our metric would be redundant. Meanwhile, such coefficient ranges were rather low, from -0.004 (in EDR) to 0.44 (in OHD).

We next turn to the performance aspects of the proposed method. One key performance parameter in the use of M-tree in our framework is the node size— as the page size increases, the number of I/O decreases and the performance thus improves up to a certain point, but as the CPU cost increases over a larger page size, the performance eventually starts to worsen after a trade-off point. To find it, we conducted additional experiments in which we observed that the performance significantly improves when the page size increases from 4 to 32 Kbytes, but plateaus after 32. We thus set the node size as 32 Kbytes in all subsequent experiments.

Next, we also validated the scalability of the proposed method over a synthetic trajectory datasets, which are generated by the network-based data generator in [31], in various experimental settings, varying the retrieval size, the length of the trajectory and the cardinality of the database. Naturally, the performance of our proposed method depends on the cost for searching M-tree, as we adopt search methods of M-tree for efficient query processing. We therefore briefly discuss the scalability of the proposed method compared with the linear scan. Our proposed method consistently outperforms the linear scan in all three types of matching queries. The speedup of our proposed method is up to 69, 5.2, 14.6 over the linear scan in the whole, subpattern, and reverse subpattern matching respectively.

Finally, to demonstrate the pruning effect of Theo-

rem 2, 3, and 4, we report the distance computation ratio per trajectory in Fig. 6. The ratio of the number of distance computation required by our proposed framework to that of linear scan is 0.2, 0.23, and 0.27 for the whole, subpattern and reverse subpattern matching respectively. Note that the ratio is the lowest for the whole pattern matching, as its pruning condition using the two-way distance is relatively stricter compared to the other matchings using only the one-way distance.

7 CONCLUSION

In this paper, we study how to support pattern matching queries for trajectories constrained by road networks. Toward this goal, we first investigate the requirements for the trajectory representation and distance measure for our target problem. As none of the existing work fully satisfies the requirements identified, we then devise the trajectory representation and the distance measure, which fulfill all the requirements.

More specifically, we study the three pattern matching queries (whole, subpattern, and reverse subpattern matching) to search for similar trajectories to the given query trajectory. Though the notion of *similarity* varies across different types of queries, we proposed a unified framework efficiently supporting range and KNN queries for all three types of matching based on M-tree and pruning rules. We validated the quality of results by visualizing the results for different types of queries over real-life road network trajectories. Comparison with other distance measures showed that the proposed distance measure is more appropriate for road network trajectories.

There are also opportunities for future research. One opportunity is to study a specialized index structure for the reverse subpattern matching, which is inherently more complex compared to other matchings, as our M-tree based framework, though achieving the speed up of 14.6 over the baseline approach, still incurs high computational cost.

ACKNOWLEDGEMENTS

This research was supported by Microsoft Research Asia and the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute for Information Technology Advancement) (NIPA-2010-C1090-1031-0009) and the Engineering Research Center of Excellence Program of Korea Ministry of Education, Science and Technology (MEST) / National Research Foundation of Korea (NRF) (Grant 2010-0001728).

REFERENCES

- [1] C. S. Jensen, D. Lin, and B. C. Ooi, "Query and update efficient b+-tree based indexing of moving objects," in *Proc. VLDB*, 2004, pp. 768–779.
- [2] M. F. Mokbel, T. M. Ghanem, and W. G. Aref, "Spatio-temporal access methods," *IEEE Data Eng. Bull.*, vol. 26, no. 2, pp. 40–49, 2003.
- [3] D. Pfoser and C. S. Jensen, "Trajectory indexing using movement constraints*," *Geoinformatica*, vol. 9, no. 2, pp. 93–115, 2005.
- [4] V. T. D. Almeida and R. H. Guting, "Indexing the trajectories of moving objects in networks," *Geoinformatica*, vol. 9, no. 1, pp. 33–60, 2005.
- [5] X. Li and H. Lin, "Indexing network-constrained trajectories for connectivity-based queries," *International Journal of Geographical Information Science*, vol. 20, no. 3, pp. 303–328, 2006.
- [6] E. J. Keogh, "Exact indexing of dynamic time warping," in *Proc. VLDB*, 2002, pp. 406–417.
- [7] K. Mouratidis, M. L. Yiu, D. Papadias, and N. Mamoulis, "Continuous nearest neighbor monitoring in road networks," in *Proc. VLDB*, 2006, pp. 43–54.
- [8] M. Vazirgiannis and O. Wolfson, "A spatiotemporal model and language for moving objects on road networks," in *Proc. SSTD*, 2001, pp. 20–35.
- [9] B. Lin and J. Su, "One way distance: For shape based similarity search of moving object trajectories," *Geoinformatica*, vol. 12, no. 2, pp. 117–142, 2008.
- [10] E. Frentzos, K. Gratsias, and Y. Theodoridis, "Index-based most similar trajectory search," in *Proc. ICDE*, 2007, pp. 816–825.
- [11] S.-L. Lee, S.-J. Chun, D.-H. Kim, J.-H. Lee, and C.-W. Chung, "Similarity search for multidimensional data sequences," in *Proc. ICDE*, 2000, pp. 599–608.
- [12] M. Vlachos, D. Gunopulos, and G. Kollios, "Discovering similar multidimensional trajectories," in *Proc. ICDE*, 2002, pp. 673–684.
- [13] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proc. SIGMOD*, 2005, pp. 491–502.
- [14] L. Chen and R. T. Ng, "On the marriage of lp-norms and edit distance," in *Proc. VLDB*, 2004, pp. 792–803.
- [15] J. Gudmundsson, M. van Kreveld, and B. Speckmann, "Efficient detection of motion patterns in spatio-temporal data sets," in *Proc. GIS*, 2004, pp. 250–257.
- [16] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen, "Discovery of convoys in trajectory databases," in *Proc. VLDB*, 2008, pp. 1068–1080.
- [17] P. Bakalov, E. Keogh, and V. J. Tsotras, "TS2-tree - an efficient similarity based organization for trajectory data," in *Proc. GIS*, 2007, pp. 1–4.
- [18] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," in *Proc. SIGMOD*, 2007, pp. 593–604.
- [19] H. Alt, C. Knauer, and C. Wenk, "Matching polygonal curves with respect to the fréchet distance," in *Proc. STACS*, 2001, pp. 63–74.
- [20] Y. Yanagisawa, J. Akahani, and T. Satoh, "Shape-based similarity query for trajectory of mobile objects," in *Proc. MDM*, 2003, pp. 63–77.
- [21] S. Brakatsoulas, D. Pfoser, R. Salas, and C. Wenk, "On map-matching vehicle tracking data," in *Proc. VLDB*, 2005, pp. 853–864.
- [22] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-Tree: An efficient and robust access method for points and rectangles," in *Proc. SIGMOD*, 1990, pp. 322–331.
- [23] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query processing in spatial network databases," in *Proc. VLDB*, 2003, pp. 802–813.
- [24] G. Klunder and H. Post, "The shortest path problem on large-scale real-road networks," *Networks*, vol. 48, no. 4, pp. 182–194, 2006.
- [25] B. V. Cherkassky, A. V. Goldberg, and T. Radzik, "Shortest paths algorithms: theory and experimental evaluation," *Math. Program.*, vol. 73, no. 2, pp. 129–174, 1996.
- [26] F. B. Zhan and C. E. Noon, "Shortest path algorithms: An evaluation using real road networks," *Transportation Science*, vol. 32, no. 1, pp. 65–73, 1998.
- [27] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: An efficient access method for similarity search in metric spaces," in *Proc. VLDB*, 1997, pp. 426–435.
- [28] http://cs.uic.edu/~boxu/mp2p/gps_data.html.
- [29] <http://www.census.gov/geo/www/tiger>.
- [30] M. Kendall and J. D. Gibbons, *Rank correlation methods*, 5th ed. A Charles Griffin Book, 1990.
- [31] T. Brinkhoff, "Generating traffic data." *IEEE Data Eng. Bull.*, vol. 26, no. 2, pp. 19–25, 2003.