

# A Road Network Embedding Technique for K-Nearest Neighbor Search in Moving Object Databases \*

Cyrus Shahabi, Mohammad R. Kolahdouzan, Mehdi Sharifzadeh  
Department of Computer Science  
University of Southern California  
Los Angeles, California 90089

[shahabi, kolahdoz, sharifza]@usc.edu

## ABSTRACT

A very important class of queries in GIS applications is the class of K-Nearest Neighbor queries. Most of the current studies on the K-Nearest Neighbor queries utilize spatial index structures and hence are based on the Euclidean distances between the points. In real-world road networks, however, the shortest distance between two points depends on the actual path connecting the points and cannot be computed accurately using one of the *Minkowski* metrics. Thus, the Euclidean distance may not properly approximate the real distance. In this paper, we apply an embedding technique to transform a road network to a high dimensional space in order to utilize computationally simple *Minkowski* metrics for distance measurement. Subsequently, we extend our approach to dynamically transform new points into the embedding space. Finally, we propose an efficient technique that can find the actual shortest path between two points in the original road network using only the embedding space. Our empirical experiments indicate that the Chessboard distance metric ( $L_\infty$ ) in the embedding space preserves the ordering of the distances between a point and its neighbors more precisely as compared to the Euclidean distance in the original road network.

## Categories and Subject Descriptors

H.2.8 [Spatial Databases and GIS]

## General Terms

Algorithms.

---

\*This research has been funded in part by NSF grants EEC-9529152 (IMSC ERC) and IIS-0082826, NIH-NLM grant nr. R01-LM07061, NASA/JPL contract nr. 961518, DARPA and USAF under agreement nr. F30602-99-1-0524, and unrestricted cash/equipment gifts from NCR, Microsoft and SUN.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GIS'02, November 8–9, 2002, McLean, Virginia, USA.  
Copyright 2002 ACM 1-58113-XXX-X/02/0011 ...\$5.00.

## Keywords

K Nearest Neighbors, Moving Objects, Road Networks, Space Embedding, Chessboard Metric.

## 1. INTRODUCTION

The K-Nearest Neighbor (KNN) queries are frequently issued in multidimensional spaces. These queries ask for the  $K$  closest points to a query point with respect to some distance function. The complexity of the selected distance function has direct impact on the complexity of these queries. These distance functions are often computationally complex because of either the nature of the function and/or the large number of dimensions. A *Road Network* is a special case of two dimensional spaces where the objects are points that are inter-connected by roads and the dimensions specify the geographical coordinates (i.e., latitude and longitude) of the points. An example KNN query for such networks is to find the  $K$  closest gas stations to a specific location. Evaluating KNN queries for such networks is computationally expensive because the distance is a function of the network paths connecting the points (e.g., shortest path between two points). Now consider an application where the query point  $q$  is moving (e.g., it is a car). In this case, the distance function  $D$  from  $q$  to the points of interest is to be computed very often and in real-time. This renders the computation of complex distance functions impractical for real-time KNN queries for moving objects.

The majority of the current research on different aspects of KNN queries are based on utilizing different spatial index structures such as R-Tree or Quad-Tree. The use of index structures for distance measurements implicitly implies the use of Euclidean distance between the points. The first contribution of this paper is that it demonstrates that the Euclidean metric is not a good distance approximation for road networks. Our experiments with real-world data show approximately 40% false hits for 100% recall when Euclidean metric is used. The experiments also show that even for lower percentages of recall, in which Euclidean metric provides a perfect 100% precision, the results are highly out of order.

Therefore, we propose a new approach for KNN queries in road networks that can address both static and moving query points. Our approach, termed Road Network Embedding (RNE), is based on transforming a road network into a higher dimensional space in which simpler distance functions

can be utilized. In addition, we show that the Chessboard distance metric in RNE provides a proper approximation of the actual distances between the points. One drawback of RNE is that the embedding technique requires an off-line pre-computation of the shortest paths between all the points in the network. However, we show that for a high percentage of the points, the Chessboard distance can be computed using only a subset of the dimensions. Hence, we introduce the notion of truncated RNE, where the shortest paths from only a small percentage of the points are computed off-line. Our experiments verify that while truncated RNE provides an acceptable accuracy, it extensively reduces the space and computation complexity of RNE.

Our proposed RNE approach is aimed for the KNN queries when the query points are static. For moving query points, RNE must re-compute the embedding of the original network whenever the object moves (i.e, a new point is added to the original space). To address this problem, we propose an extension to RNE, termed D-RNE, to dynamically embed moving or new query points. Finally, while RNE can find the nearest neighbor, it cannot determine the actual path between that neighbor and the query point. Hence, we propose a greedy-heuristic algorithm, termed SP-RNE, to find the actual path between the points using only their transformations in RNE.

The remainder of this paper is organized as follows. In Section 2, we review the current research on the KNN problem. Section 3 discusses the problem of KNN in road networks and briefly describes a naive solution based on pre-computation of all the distances off-line. Section 4 provides a background on the embedding techniques. We describe our approach for KNN queries by embedding the road networks into higher dimensional spaces, and compare it with the naive pre-computation approach in Section 5. In Section 6, we propose two techniques for dynamic embedding of moving objects and finding the shortest path using the embedding space. We discuss our experimental results and future work in Sections 7 and 8, respectively.

## 2. RELATED WORK

Recently, many new techniques have been proposed for the K-Nearest Neighbor and range queries in two and multi-dimensional spaces that can be adapted to road networks. They can be categorized into two groups. The first group partitions the space by utilizing different spatial or conventional index structures such as R-Tree [3] and its variants [2]. The second group, graph-based, are based on pre-computation of the nearest neighbors, and then use of index structures and/or Voronoi diagram.

As examples of the first group, Roussopoulos et al. in [6] propose branch-and-bound R-tree traversal algorithm to find the nearest neighbor objects to a point, and then generalize it to find the  $k$  nearest neighbors. The main drawback of their algorithm is the depth-first traversal of the index tree that incurs unnecessary disk accesses. Hjaltason and Samet in [4] propose a general incremental nearest neighbor algorithm that uses a priority queue on the index tree to reduce disk accesses. Their algorithm is adapted to R-Tree and is suitable for distance browsing queries but does not provide a substantial improvement in performance over the current R-Tree based KNN algorithms. Yufei et al. in [7] propose query processing methods that use R-Tree as the underlying data structure to address nearest neighbors for a query

point that is moving on a straight line segment. In general, the use of index structures (e.g., R-Tree, Quad-Tree) for distance measurement implicitly implies use of Euclidean distance between the objects. This may not necessarily be a good approximation of the actual distance between objects in a road network where the distance between two points depends on the actual path connecting the points and cannot be computed accurately using one of the *Minkowski* metrics.

In [8], Yu et al. propose partitioning the data in a high-dimensional space and selecting a reference point for each partition. The data in each partition are transformed into a single dimensional space based on their similarity with respect to a reference point, and KNN queries are performed using one-dimensional range search on a  $B^+$ -Tree index. The effectiveness of this approach depends on how the data are partitioned. In [1], Berchtold et al. propose algorithm for similarity search in multimedia databases with large set of high-dimensional points. They suggest pre-computing the result of any nearest-neighbor search, that corresponds to a computation of the Voronoi cell of each data point and storing the Voronoi cells into an index structure. The nearest neighbor query is then equivalent to finding the Voronoi diagram that contains the query point. Even though the voronoi diagram techniques are efficient for first nearest neighbor queries, but their extension for the KNN queries requires a priori knowledge of the value of  $K$ .

The proposed approach in this paper can be categorized in the second group and is based on transforming the road networks into a higher dimensional space. The transformation requires a pre-computation of the distances from all or a group of points to all other points.

## 3. PROBLEM DEFINITION

Consider a set of  $n$  multidimensional objects  $S = (o_1, o_2, \dots, o_n)$  and a function  $D$  that specifies the distance between the objects. The  $K$  nearest neighbors problem with respect to a *query point*,  $q$ , is to find a set  $S' \subseteq S$  of  $K$  objects with smallest distances to  $q$ , that is for any object  $o' \in S'$  and  $o \in S - S'$ ,  $D(o', q) \leq D(o, q)$ . The distance function  $D$  usually requires expensive operations. For example, in a biological database that contain information about protein molecules, a distance function performs complicated quadratic-time structural comparisons to find the similar molecules.

In the context of road networks and moving objects, the *original space* contains two-dimensional objects: intersections (*original nodes*) connected by some streets. The query points in such spaces are usually moving objects (e.g., cars) travelling through the streets from a source to a destination and the KNN problem is defined as finding the closest *points of interest* (e.g., hospitals, gas stations) to the moving objects. Some of the challenges in such scenarios are:

- The distance function  $D$  between two original nodes in the road networks is usually specified as the length of the path between the nodes with some minimum weight (e.g., time to travel along the path). These weights result in complex algorithms for computation of distance functions (e.g., Dijkstra algorithm to find the minimum weighted path in a network with complexity  $O(e + n \log n)$ , where  $e$  and  $n$  are number of edges and nodes in the network respectively).
- When the query point  $q$  is a moving object, the distance function  $D$  from  $q$  to the points of interests is to

be computed very often and in real-time. This renders the computation of complex distance functions impractical for real-time KNN queries for moving objects.

There are three different ways to address these challenges:

1. One approach is to approximate the complex distance function  $D$  with a simpler distance function  $D'$  in the same original space. For example, the shortest path between two intersections in a road network can be approximated by their Euclidean distance. The advantage of this approach is that the well studied spatial index structures that are based on the Euclidean distance can be utilized to address regular and constraint KNN queries. The disadvantage, however, is that the Euclidean distance does not properly approximate the actual distance function in the road networks.
2. Another naive approach is the *pre-computation* of all shortest distance pairs during an off-line process. Consider a road network  $S$  with  $n$  intersections and  $e$  roads. In this approach, we pre-compute the shortest distances between all intersections of  $S$  and store the distances into a database. Hence, this approach addresses the challenge of real-time KNN queries by retrieving the distance between two points from the database rather than computing it real-time. The space complexity of this approach is  $O(n^2)$  while the computation complexity is  $O(n(e + n \log n))$ .
3. We propose a third approach as to introduce a new multi-dimensional space with simple distance functions  $D'$ s (e.g., *Minkowski* distance metrics) that can either precisely approximate the distances between the original nodes (i.e.,  $D'(o', p') = D(o, p)$ ), or at least preserve the ordering of the distances between the original nodes (i.e.,  $D(p, o) \leq D(p, q) \Rightarrow D'(p', o') \leq D'(p', q')$ ).

In Section 5 we discuss our proposed approach to transform the road network into an embedding space. We compare the pros and cons of approaches 2 and 3 in Section 5.2.

## 4. BACKGROUND ON EMBEDDING

In this section, we provide an overview of the *space embedding* techniques. Let  $(S, D)$  be a *finite metric space* where  $S$  is a finite set of  $n$  objects and  $D : S * S \rightarrow \mathbb{R}^+$  is a distance metric over  $S$ . The *embedding*, or *transformation*, of a finite metric space  $(S, D)$  into a *vector space*  $(\mathbb{R}^k, D')$  is a mapping  $E : S \rightarrow \mathbb{R}^k$  where  $k$  is the dimensionality of the vector space and  $D'$  is one of *Minkowski*  $L_p$  metrics in  $\mathbb{R}^k$ :

$$L_p(x, y) = \left[ \sum_{i=1}^k |x_i - y_i|^p \right]^{1/p} \quad (1)$$

in which  $x_i$  and  $y_i$  are the  $i$ th coordinates of two points  $x, y$  in space respectively, and  $p$  is the order of the Minkowski metric. The first order of the Minkowski metric,  $L_1$ , is known as *Manhattan distance*, the second order  $L_2$  as *Euclidean distance*, and the infinite order  $L_\infty = \max_i |x_i - y_i|$  as *Chessboard distance*. The objective of embedding is to have a fast and computationally simple  $D'$  function such that  $D(x, y) \cong D'(E(x), E(y))$ . In other words, the distance between two objects in the original metric space should be close enough to the distance between their corresponding embedded points in the embedding space. The quality of an embedding technique  $E$  is measured by *distortion* and *stress*. Distortion specifies the maximum difference between distance functions  $D$  and  $D'$  and is equal to  $c_1 \times c_2$  ( $c_1, c_2 \geq 1$ )

when it is guaranteed that for an embedding technique  $E$ :

$$\frac{D(x, y)}{c_1} \leq D'(E(x), E(y)) \leq D(x, y) \cdot c_2 \quad (2)$$

for all pairs of objects  $x, y \in S$ . Stress represents the overall deviation in the distance and is defined as:

$$Stress = \frac{\sum_{x, y \in S} (D'(E(x), E(y)) - D(x, y))^2}{\sum_{x, y \in S} D(x, y)^2} \quad (3)$$

The optimum  $D'$  function generates zero stress, equivalent to no distortion (i.e.,  $c_1 = c_2 = 1$ ). An embedding technique  $E$  is *contractive* when  $D'(E(x), E(y)) \leq D(x, y)$  and *proximity preserved* when for all  $x, y$  and  $z$  in the original space,  $D(x, y) \leq D(x, z)$  can be concluded from  $D'(E(x), E(y)) \leq D'(E(x), E(z))$ .

In Section 5, we describe how we transform a road network to an embedding space using *Lipschitz* embedding technique.

## 5. ROAD NETWORK EMBEDDING (RNE)

A road network can be modelled as a weighted graph  $G = (V, E)$ . Let  $|V| = n$  be the number of nodes in  $G$  (i.e., road intersections),  $|E| = m$  be the number of edges in  $G$  (i.e., roads), and  $W(e)$  be the weight of an edge  $e \in E$  (e.g., length of a road). The distance  $d(u, v)$  from node  $u$  to  $v$  is defined as the length of the minimum weighted path from  $u$  to  $v$ . We assume that for arbitrary nodes  $u, v$  and  $w$ ,  $G$  is undirected:  $d(u, v) = d(v, u)$ , and  $W(e)$  is defined such that the triangular inequality holds:  $d(u, v) \leq d(u, w) + d(w, v)$ . Therefore, the set of nodes in  $G$  with the distance function  $d$  generates a metric space  $(V, d)$ . In this space,  $d$  is symmetric, non-negative and obeys triangular inequality.

We propose to utilize *Limial, London and Robinovich (LLR)* [5] embedding technique on road networks. *LLR embedding* technique is a contractive specialization of *Lipschitz* embedding, in which an object in the original space is mapped to a point in a  $k$  dimensional vector space. Consider space  $(S, D)$  in which  $D(\cdot, \cdot)$  is a distance function between the objects in  $S$ . Distance  $D$  is extended as follows: let  $S_i$  be a subset of  $S$  and  $D(x, S_i) = \min_{y \in S_i} \{D(x, y)\}$ , that is,  $D(x, S_i)$  is the distance from  $x$  to its closest neighbor in  $S_i$ . Let  $R = \{S_1, S_2, \dots, S_k\}$  be a set of subsets of  $S$ . Lipschitz embedding with respect to  $R$  is then defined as:  $E(x) = [D(x, S_1), D(x, S_2), \dots, D(x, S_k)]$ , which is a  $k$ -dimensional point in a vector space with each axis corresponding to a reference set in  $R$ . LLR embedding defines  $R$  as a set of  $O(\log^2 n)$  subsets of  $S$ :  $R = \{S_{1,1}, \dots, S_{1,\kappa}, \dots, S_{\beta,1}, \dots, S_{\beta,\kappa}\}$  where  $\kappa = O(\log n)$  and  $\beta = O(\log n)$ . Thus the original space is embedded into a  $O(\log^2 n)$  dimensional space. Each subset  $S_{i,j}$  is defined as a random subset of  $S$  with size  $2^i$ . This means that the first  $\kappa$  reference sets have 2 objects, the second  $\kappa$  reference sets have 4, etc., until the last  $\kappa$  reference sets that have approximately  $n/2$  objects. The embedding  $E(x)$  defined above for LLR has a distortion of  $O(\log n)$  (i.e.,  $c_1 = 1$  and  $c_2 = O(\log n)$ ). By using LLR technique, metric space  $(V, d)$  which represents the road network is embedded into  $(\mathbb{R}^k, d')$  in which  $k = O(\log^2 n)$  and  $d'$  distance function is one of the  $L_p$  distance metrics over  $\mathbb{R}^k$  as defined in Equation 1. Hence each node  $v$  in the original network is mapped to the point  $E(v)$  in  $O(\log^2 n)$  dimensional embedding space:

$$E(v) = (E_{S_{1,1}}(v), \dots, E_{S_{1,\kappa}}(v), \dots, E_{S_{\beta,1}}(v), \dots, E_{S_{\beta,\kappa}}(v)) \quad (4)$$

in which  $E_{S_{i,j}}(v) = d(v, S_{i,j})$ .

## 5.1 RNE: Making LLR Practical

Although LLR is a good start for transformation of road networks, it needs some tweaking to render it practical.

- To reduce the computation complexity of RNE, we introduce the notion of *Truncated Embedding Space* as an embedding space where only the distance between the original nodes and the first few reference sets are computed and considered as the attributes of the embedded points. The intuition is as follows. As discussed in Section 5, the value of the  $(i, j)$ th dimension of the embedded point  $E(x)$  is defined as the minimum distance between the original node  $x$  and the  $(i, j)$ th reference set in  $R$ . We define  $A$  and  $B$  as sets of subsets of  $S$ :

$$A = (S_{1,1}, \dots, S_{1,\kappa}, \dots, S_{p,q}) \quad (5)$$

$$B = (S_{p',q'}, \dots, S_{p',\kappa}, \dots, S_{\beta,1}, \dots, S_{\beta,\kappa}) \quad (6)$$

in which  $1 \leq p \ll \beta$  and  $1 \ll p' \leq \beta$ . The sets  $A$  and  $B$  contain the first and the last few reference sets of  $R$ , respectively. Since the number of nodes in the reference sets of  $A$  are far less than the number of nodes in the reference sets of  $B$  (e.g., sets with  $p = 1$  contain 2 nodes while the sets with  $p' = \beta$  contain  $n/2$  nodes), the distances between original nodes  $x$  and  $y$  to the sets in  $A$  are probably greater than the distances to the sets in  $B$ . Hence, the values of  $|E_{i,j}(x) - E_{i,j}(y)|$  that correspond to the reference sets in  $A$  are probably greater than the values corresponding to the sets in  $B$ . This means that the first dimensions of the embedded points, and hence the first reference sets of  $R$ , are more effective than the last dimensions when the Chessboard metric is used. The results of our experiments confirm this intuition.

- The Chessboard metric can precisely approximate the actual distance between some nodes (i.e.,  $d' = d$ ). Consider the contractive property of the LLR approach that implies when  $(V, d)$  is embedded into  $(\mathfrak{R}^{\log^2 n}, L_\infty)$ :

$$d'(E(x), E(y)) = \max_{i,j} |d(x, S_{i,j}) - d(y, S_{i,j})| \leq d(x, y) \quad (7)$$

This means that when the Chessboard metric is used for distance measurement,  $d'$  is always less than or equal to  $d$ . The equality of  $d'$  and  $d$  holds in the following two cases:

1. When for some  $i, j$ ,  $S_{i,j} \in R$ ,  $x \in S_{i,j}$  and  $x$  is the nearest node to  $y$  in  $S_{i,j}$ . Therefore  $d(y, S_{i,j}) = d(y, x)$  and  $d(x, S_{i,j}) = 0$ . Hence  $|d(x, S_{i,j}) - d(y, S_{i,j})| = d(x, y)$ .
2. When for some  $i, j$ ,  $S_{i,j} \in R$ , the shortest path from  $x$  to its nearest node in  $S_{i,j}$ , say  $z$ , passes through  $y$ . In this case  $z$  is also the nearest node to  $y$  in  $S_{i,j}$ . Therefore  $d(x, S_{i,j}) - d(y, S_{i,j}) = d(x, z) - d(y, z) = d(x, y)$ .

For the original nodes that are close to each other, the Chessboard metric provides a better approximation for the actual distance ( $d' \simeq d$ ). The intuition is as follows. Consider two original nodes  $x, y$  that are very close to each other. Also suppose that the Chessboard distance between  $x$  and  $y$  is computed using reference set  $S_{i,j} \in S$  which as stated above, is probably one of the first few reference sets of  $S$  with only a few nodes. These assumptions imply that: a) one node,  $z$ , in  $S_{i,j}$  is probably the closest node to both  $x$  and  $y$ ,

and b)  $z$  is far from  $x$  and  $y$ . These resemble a triangle  $\triangle xzy$  where the  $\widehat{xzy}$  angle is close to  $0^\circ$  and hence  $|d(x, z) - d(y, z)| \simeq d(x, y)$ .

Note that in the computation of the Chessboard distance, different  $S_{i,j}$ s are evaluated. The combination of  $x, y$  and each  $S_{i,j}$  resembles a different triangle, but the one that has the minimum value for angle  $\widehat{xzy}$ , and hence leads to the maximum value for  $d'$ , is picked by the Chessboard metric.

## 5.2 Analysis

In this section, we compare RNE with the naive pre-computation approach discussed in Section 3 in terms of precision, storage requirement, computation complexity and functionality.

**Precision:** As discussed in Section 3, the naive approach is based on pre-computing the shortest path between all pairs of the original nodes. Hence this approach provides a 100% precision when the distance between two original nodes are requested. In contrast, RNE provides an approximation of the actual distance which in the worst case is distorted by a factor of  $O(\log n)$ . Our experiments, however, show that for real world road networks, distortion is usually far less than the worst case and hence RNE also provides acceptable precision in practice.

**Storage Requirement:** The space complexity of the pre-computation approach is  $O(n^2)$  since the shortest path between all pairs of  $n$  original nodes generate a symmetric matrix with  $n^2$  elements, equivalent to  $n^2/2$  tuples in a database. In contrast, each node in RNE is mapped to only one point (i.e.,  $n$  tuples in the database) with  $\log^2 n$  dimensions, requiring space complexity of  $O(n \log^2 n)$ . The space complexity for truncated RNE is  $O(n\sqrt{n})$ .

**Computation Complexity:** RNE's off-line computation is more computationally complex than the pre-computation approach since it uses the original distance function  $d$  (e.g., shortest travelling time in a road network) to compute the transformations of the original nodes. However, truncated RNE requires the shortest path computation to  $(O(\sqrt{n}))$  of the original nodes, and provides a better computation complexity:  $O(n^2\sqrt{n})$  as compared to  $O(n^3)$  for the pre-computation approach.

**Functionality:** RNE is complementary to the current research on different aspects of the K nearest neighbor problem and embedding techniques. For example, caching techniques for querying K nearest neighbors of moving objects that are based on Euclidean distance and R-Tree index structure can be adapted to be used in the embedding space by using Chessboard distance and the X-Tree [2] index structure. In contrast, the pre-computation approach only keeps the distances between all pairs of the original nodes and hence no index structure can be utilized to explore the geometry features of the nodes.

## 6. RNE EXTENSIONS

The RNE approach discussed in Section 5 is neither aimed to address the points that dynamically change location (e.g., moving query points), nor can it identify the actual path between the points. In this section, we propose two extensions to RNE in order to: a) dynamically embed moving objects, and b) find the actual path between two points in the original space using their transformations in the embedding space.

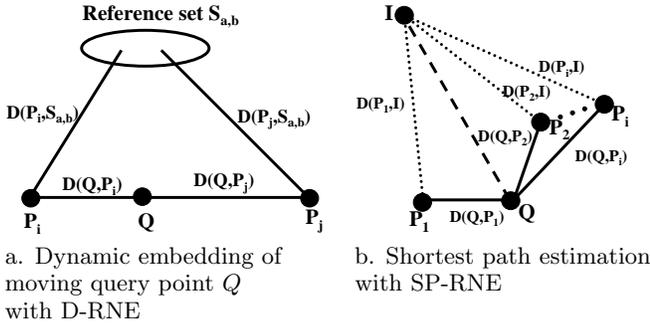


Figure 1: RNE Extensions

## 6.1 Dynamic RNE (D-RNE)

In the road network embedding approach discussed so far, we assumed that the query points and the points of interest are subsets of the original points (i.e., intersections) and are embedded off-line. This is not a realistic assumption since the query points are often moving objects and the location of the points of interest are usually between intersections and may not even be predetermined. With the current embedding techniques, insertion of a new set of points into the original space leads to the recomputation of the transformations of all original points in the initial original space. This renders these techniques impractical for real-time K nearest neighbors for moving objects. Hence an online embedding technique is required to embed these points as their locations change.

We propose an extension to the transformation technique discussed in Section 5 to embed the query points real-time. Our technique utilizes two features:

1. The query points and points of interest in a road network are always on the paths between the original nodes, meaning that insertion of these points into the original space does not introduce new edges into the original graph and does not change the distances between the original nodes.
2. The probability of using those subsets of  $S$  that have fewer number of nodes is higher with the Chessboard metric  $L_\infty$  as the distance measure (as discussed in Section 5.1).

Consider Figure 1a where the query point  $Q$  is on the path between the directly connected original nodes  $P_i$  and  $P_j$ . From feature 1, we can conclude that if the query point  $Q$  was initially in the original space, the distance between  $Q$  and  $S_{a,b}$ , the  $(a, b)$ th dimension of  $E(Q)$ , would be calculated as:

$$E_{S_{a,b}}(Q) = D(Q, S_{a,b}) = \min(D(Q, P_i) + D(P_i, S_{a,b}), D(Q, P_j) + D(P_j, S_{a,b})) \quad (8)$$

Insertion of  $Q$  after the original nodes are embedded may change some of the subsets of  $S_{a,b}$  with a probability of  $2^a/n$ . This probability is negligible for the road networks with large number of original nodes ( $n \gg 1$ ) and the subsets  $S_{a,b}$  with a very few nodes ( $a \gg 1$ ). This means that inclusion of  $Q$  into the original space would not have changed the subsets of  $S$  that have a few nodes, but may have changed the subsets that contain large number of nodes (i.e.,  $a \gg 1$ ), which in turn may lead to changes in the values of  $(a, b)$ th dimensions of the embedded points. However, feature 2 indicates that these dimensions are not effective anyway when the Chessboard metric is used for distance measurements.

We conclude that despite possible changes in the embedding of the original nodes when a new node is inserted into the original space, the Chessboard distances between the embedded points still remain the same. Hence we generalize Equation 8 to calculate all dimensions of  $E(Q)$ .

## 6.2 Shortest Path in RNE (SP-RNE)

While the embedding techniques are intended to approximate the actual complex distance functions with simpler functions, they are not aimed to find the shortest path between the nodes in a road network. In this section, we propose a greedy-heuristic algorithm to find the shortest path between two points in the original space using only the distances between their transformations in the embedding space. Consider the query point  $Q$  and point of interest  $I$  in Figure 1b and suppose that  $Q$  is only connected to points  $P_1, P_2, \dots, P_i$ . The intuition of our algorithm is that for a perfect embedding function  $E$  that  $D(x, y) = D'(E(x), E(y))$  for all  $x, y \in S$ , if the shortest path from  $Q$  to  $I$  passes through points  $\rho_1, \rho_2, \dots, \rho_i$ , then the shortest path from  $E(Q)$  to  $E(I)$  passes through  $E(\rho_1), E(\rho_2), \dots, E(\rho_i)$  and vice versa. We assume that for an embedding function with distortion, the probability that the shortest path from  $E(Q)$  to  $E(I)$  does not pass through  $E(\rho_1), E(\rho_2), \dots, E(\rho_i)$  is proportional to the distortion: the higher the distortion, the higher the possibility that the shortest path from  $E(Q)$  to  $E(I)$  goes through different points than the shortest path from  $Q$  to  $I$ .

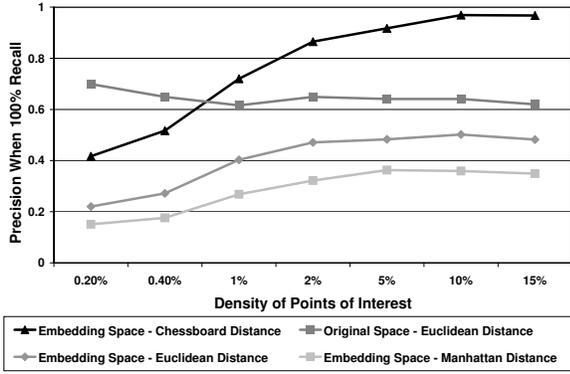
The algorithm tries to find the global optimum solution by searching through the local optimums. In the first step, a point  $P_j$  among all directly connected neighbors of the query point  $Q$  is found such that the length of the path from the transformation of  $Q$  to  $I$  through  $P_j$  is minimum. This point is selected as the next point in the shortest path from  $Q$  to  $I$ , and is then considered as the next query point. The algorithm continues on the next query points until it reaches a point that is directly connected to  $I$ .

The complexity of the algorithm is proportional to the length of the diameter of the network: the longest path between any two points in the network when  $W(e) = 1$  for all  $e$ . The upper-bound of the complexity is  $O(n)$  but in a real world road network that nodes have a degree of 4 or less, the complexity of the algorithm is far less. For example, the average complexity of the algorithm for a Manhattan network is  $O(\sqrt{n})$ .

## 7. PERFORMANCE EVALUATION

We conducted several experiments to: 1) compare the precision of different Minkowski metrics for distance measurement in the embedding space discussed in Section 4, 2) study the impact of K and density of the points of interest on the performance of the RNE approach, and 3) study the accuracy of D-RNE approach and SP-RNE technique discussed in Section 6.

For our experiments, we used a real data set for Kuwait obtained from NavTech Company. The data covers a rectangular area with corner points latitude and longitude (47.51, 29.06) and (48.44, 29.60) and contains 117,000 road segments, that constitute a graph with approximately 50,000 nodes. Different features in that area were also used as points of interest with different densities (density of the points of interest is defined as the number of points of interest over the number of original nodes). We use the precision-recall

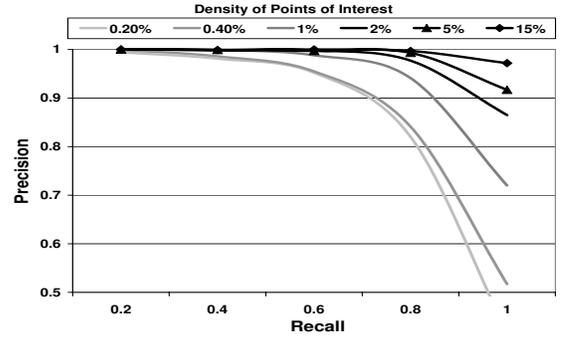


**Figure 2: Precision comparison of different distance metrics with  $K=5$**

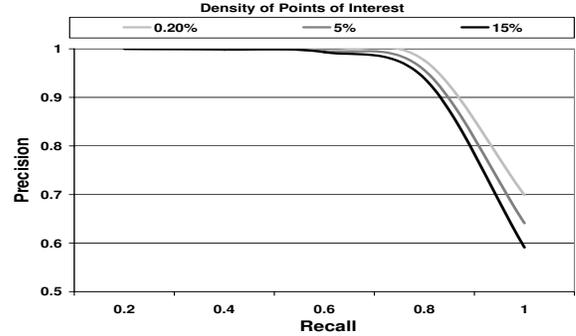
metric to measure the accuracy and the longest common subsequence to measure how precisely each approach/metric preserves the ordering in the result sets.

In our first set of experiments, we investigated how precisely different metrics for distance (i.e., Chessboard, Euclidean and Manhattan) in the embedding space approximate the real distance in the original space, by finding the  $K$  nearest neighbors of the query points when the density of the points of interest varies. Figure 2 depicts the precision of these metrics when  $K=5$  and 100% recall is achieved (i.e., when all the correct nearest neighbors are found). Our experiments for other values of  $K$  show similar trends. As shown in the figure, precision of the Chessboard metric in the embedding space increases up to 98% as the density grows. In contrast, Euclidean metric in the original space behaves independent from the density and provides an almost constant precision of 60% to 70%. This means that when the Euclidean distance is used in the original space to find the  $K$  Nearest Neighbors of a query point, 40% of the result set are false hits. The figure suggests a threshold value for the density of the points of interest, 0.5% for our data set, that can be used in a query optimizer to utilize either the Euclidean metric in the original space or the Chessboard metric in the embedding space for distance measurement. The figure also shows that the Chessboard metric always outperforms Manhattan and Euclidean metrics in the embedding space. Hence, from now on, we focus on comparison between the Chessboard metric in the embedding space and the Euclidean metric in the original space.

Figure 3 depicts the precision-recall graph of the Chessboard metric in the embedding space and the Euclidean metric in the original space. As shown in Figure 3a, Chessboard metric provides a better performance as the density of the points of interest increases. This is because the higher the density of the points of interest, the higher the possibility of the first  $K$  of those points being closer to the query point, and hence less distortion (as discussed in Section 5.1). In contrast, Figure 3b shows that the performance of the Euclidean metric in the original space slightly degrades as the density increases. The intuition here is that when the points of interest are sparse, the neighbors are far enough from the query point that even the Euclidean metric can preserve the ordering of the distances. Comparison between Figure 3a and 3b also shows that the Euclidean distance in the original space provides a better precision as compared to the Chessboard metric in the embedding space only when the



a. Embedding space, Chessboard distance



b. Original space, Euclidean distance

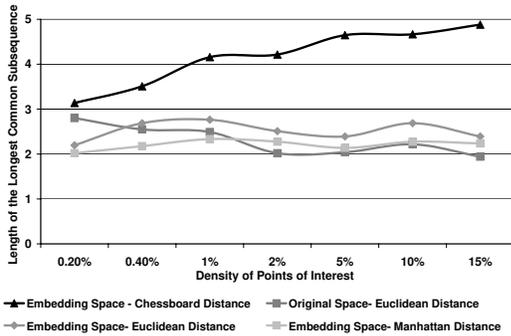
**Figure 3: Precision-Recall with  $K=5$**

points of interest are very sparse.

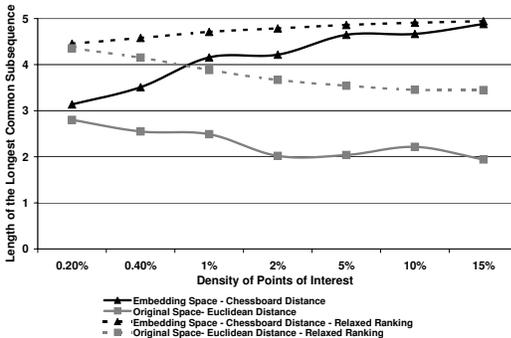
Figure 4a depicts how different approaches preserve order in the result set when  $K=5$  (results for other values of  $K$  have similar trends). The Y axis in the figure is the length of the longest subset of the result set that has the same order as the actual result set. As shown in the figure, the Chessboard metric in the embedding space always outperforms other metrics/approaches and provides better ordering in the result set as the density of the points of interest increases. In contrast, the ordering in only 40% to 60% of the results are preserved when the Euclidean metric in the original space is used. This means that even for lower percentage values of recall in which Euclidean metric provides a 100% precision (Figure 3b), the results are highly out of order. Since in certain applications, the mis-ordering of some points in the result set (i.e., the neighbors with similar distances to the query point) may be tolerable, a relaxed measure can be used to compute the longest common subsequence in the result sets. Figure 4b shows the results when we relaxed the longest common subsequence measure by 10% (i.e., we neglected the mis-orderings between the neighbors that have less than 10% difference in their distances from the query point). As shown in the figure, this small relaxation of our measure leads to much better order preservation in the result set of both Chessboard metric in the embedding space and Euclidean metric in the original space. This means that the mis-orders introduced by both metrics are mostly for the points with similar distances to the query point.

Our next set of experiments<sup>1</sup> were aimed to investigate the impact of  $K$  on the performance of the different metrics. The results show that both Chessboard and Euclidean dis-

<sup>1</sup>Due to lack of space we eliminated the details and graphs of the remaining experiments. Please refer to the full paper for details.



a. RNE vs. original space



b. RNE vs. original space, relaxed ranking

**Figure 4: Comparison of order preservation by different approaches with  $K=5$**

tance metrics perform better for smaller values of  $K$  and the performance of both decrease as the percentage of the recall increases. The reason is that the higher values of recall are achieved when the points of interest that are further from the query point are found. Higher values of  $K$  also introduce points of interest that are further from the query point. As we discussed in Section 5.1, the points of interest that are far from the query point contribute to more distortion and hence less precision.

In our next set of experiments, we synthetically generated 3500 new nodes (7% of the original nodes) in the original space to investigate the performance of D-RNE. Note that the inclusion of the new nodes does not change the distances between the original nodes. The results showed that the difference between the precisions when the synthetic nodes are embedded off-line and when they are dynamically embedded in real-time using D-RNE approach is always less than 0.8%. In our next set of experiments, we studied the accuracy of the SP-RNE technique. Our experiments showed that in 72% of the cases, the path between 2 nodes calculated by SP-RNE matches their actual shortest path, and for the other 28% is on average 11% longer.

Our final set of experiments were aimed to study the space requirements of pre-computation, RNE and truncated RNE approaches. Table 1 shows the observations. As shown in the table, 256 dimensions are required for the RNE, while the Chessboard distance is computed for over 95% and 90% of the embedded points using only the first 70 and 40 dimensions, respectively. The table also illustrates that the shortest path computation in truncated RNE approaches are performed for far less number of nodes as compared to

Approach	Dimensions	Shortest paths computed for	Number of Tuples	Required Space
Pre-Computation	-	50,000 nodes	1.25 billion	30 GB
RNE	256	50,000 nodes	50,000	103 MB
T-RNE, 95% acc.	70	672 nodes	50,000	28 MB
T-RNE, 90% acc.	40	160 nodes	50,000	16 MB

**Table 1: Complexity comparison of pre-computation, RNE and truncated RNE**

the regular RNE and pre-computation approaches: 672 and 160 nodes versus 50,000 nodes. Finally, the number of tuples generated and the total disk space required by the pre-computation approach are significantly larger than those of the RNE approaches.

## 8. CONCLUSION AND FUTURE WORK

In this paper, we focused on the class of  $K$  nearest neighbor (KNN) queries for moving objects in road networks. We proposed to apply an embedding technique to a road network (RNE) in order to convert its points to a higher-dimensional space and use the Chessboard metric for distance measurements in the new space. We introduced the notion of truncated-RNE, that has less computation complexity than RNE but still provides an acceptable precision. We also proposed two extensions to RNE: 1) D-RNE to dynamically embed new points into the embedding space and 2) SP-RNE to find the shortest path between points in the original road network using their transformations in the embedding space.

We plan to extend this study in two ways. First, we plan to modify the current Euclidean-based caching techniques for KNN queries to work for the Chessboard metric in the embedding space. Second, we intend to formalize the trade-offs between the Euclidean metric in the original space and the Chessboard metric in the embedding space in order to utilize these trade-offs within a query optimizer for choosing one approach over the other.

## 9. REFERENCES

- [1] S. Berchtold, B. Ertl, D. A. Keim, H.-P. Kriegel, and T. Seidl. Fast nearest neighbor search in high-dimensional spaces. In *Proc. 14th IEEE Conf. Data Engineering, ICDE*, 23–27, 1998.
- [2] S. Berchtold, D. A. Keim, and H.-P. Kriegel. The X-tree: An index structure for high-dimensional data. In *Proceedings of the 22nd International Conference on Very Large Databases*, pages 28–39, San Francisco, U.S.A., 1996. Morgan Kaufmann Publishers.
- [3] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD’84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18–21, 1984*, pages 47–57. ACM Press, 1984.
- [4] G. R. Hjaltason and H. Samet. Distance browsing in spatial databases. In *ACM Transactions on Database Systems*, volume 24, pages 265–318, 1999.
- [5] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. In *IEEE Symposium on Foundations of Computer Science*, pages 577–591, 1994.
- [6] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, San Jose, California, May 22–25, 1995*, pages 71–79. ACM Press, 1995.
- [7] Y. Tao, D. Papadias, and Q. Shen. Continuous nearest neighbor search. In *Proceedings of the Very Large Data Bases Conference (VLDB), Hong Kong, China, 2002*.
- [8] C. Yu, B. C. Ooi, K.-L. Tan, and H. V. Jagadish. Indexing the distance: An efficient method to KNN processing. In *The VLDB Journal*, pages 421–430, 2001.