社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2003－DBS－131　(62)
2003／7／17

Temporal Continuous Nearest Neighbor Search:

Integration of Temporal and Spatial Spaces on Road Search

馮 鈞

名古屋大学大学院工学研究科 情報工学専攻 〒 464-8603 名古屋市千種区不老町
E-mail: feng@watanabe.nuie.nagoya-u.ac.jp

渡邉 豊英

名古屋大学大学院情報科学研究科　社会システム情報学専攻 〒 464-8603 名古屋市千種区不老町
E-mail: watanabe@is.nagoya-u.ac.jp

あらまし
本稿では、交通コストを道路ネットワークの下で統合的に管理し、活用する方法を提案する。この方法では交通コストと交通制限情報が道路ネットワークのノードで管理され、道路情報を管理している空間インデックスに影響することなく, 更新される。この柔軟で適応性に富んだ表現方法によって、交通コストに基づいた連続的な最隣接目的地の検索は効率的に実現される。

キーワード　高度道路交通システム (ITS), 交通情報, 道路ネットワーク, 連続的な最隣接目的地検索

Temporal Continuous Nearest Neighbor Search:

Integration of Temporal and Spatial Spaces on Road Search

Jun FENG

Department of Information Engineering, Graduate School of Engineering, Nagoya University.
Furo-cho, Chikusa-ku, Nagoya 464-8603, JAPAN
E-mail: feng@watanabe.nuie.nagoya-u.ac.jp

Toyohide WATANABE

Department of Systems and Social Informatics, Graduate School of Information Science, Nagoya University.
Furo-cho, Chikusa-ku, Nagoya 464-8603, JAPAN
E-mail: watanabe@is.nagoya-u.ac.jp

Abstract　An important part of Intelligent Transportation System (ITS) is a geographic database containing road maps, map entities, and current travel cost on segments of transportation networks. In this paper, we center on an integrated representation of traffic cost information and spatial road network. Our method is flexible for connecting traffic cost and constraint to static road map. A query on datasets created by our method, such as continuous nearest neighbor search based on traffic cost, can be realized efficiently by taking advantages of this integrated representation.

Keywords　Intelligent Transportation System (ITS), traffic information, road netwrok, continuous nearest neighbor search

1 Introduction

An important part of Intelligent Transportation System (ITS) is a geographic database containing static map data (including data of road network and other map objects), public transportation routes, and current travel cost (e.g., travel time) on segments of transport network, which is updated frequently [1]. Queries in ITS applications are often based on the current travel-time, congestion, restrictions and other attributes of transportation network.

In this paper, we propose a representation method for integrating traffic information and road network. A query on this dataset, such as a temporal continuous nearest neighbor (CNN) search, can be realized efficiently by taking advantages of the integrated representation. The temporal CNN search retrieves the nearest target objects for every point on a predefined route on road networks based on the current traffic conditions. The result is a set of quadruples $< target, interval, path, cost >$, such that *interval* is a sub-route, *target* is the nearest target object from all the points on *interval*, and *path* is the lowest *cost* path from *interval* to *target*. This kind of search may refer to a road network in a wide area where the predefined route crosses, while the nearest neighbor (NN) search for the intervals on the route may be done on a relatively small area based on the current transportation conditions.

This paper is organized as follows. The related works for information management of road networks and our previous works on CNN search are presented in Section 2. The representation method for integrated management of temporal traffic conditions and spatial information about road network is proposed in Section 3. Section 4 introduces a reverse search method for CNN search based on the traffic cost. Section 5 analyses our method and makes a conclusion on our work.

2 Related and Previous Work

The issue of this paper refers to the integrated management method of temporal traffic conditions and spatial information about road networks, and the continuous nearest neighbor search.

2.1 Continuous nearest neighbor search

The existing work for CNN search is almost presented from the computational geometry perspective [2, 3, 4]. Their CNN search methods for line segments are effective. However, all the works are based on the straight-line distance between objects.

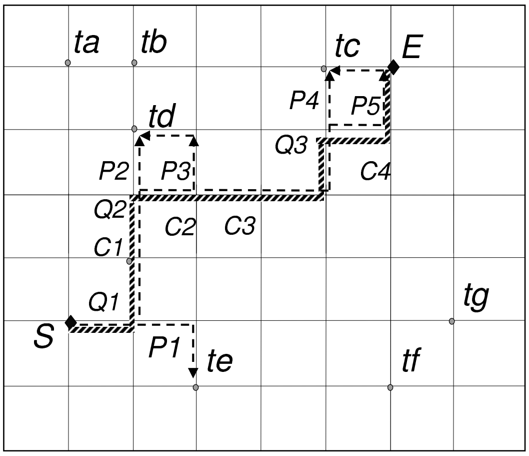


Figure 1: Road network: specific route [*S, E*] and target objects.

We have proposed the method [5] to solve the problem based on common situations in GIS: the distance from a point on the route to a target place should be decided by the path length; and the target objects and the road network are managed in GIS datasets, respectively. Consider the example given in Figure 1, where the specific route is $[S, E]$, and the target object set is $\{t_a,\ t_b,\ t_c,\ t_e,\ t_f,\ t_g\}$. The output of the query is $\{< t_e, [S, Q_1], P_1 >,\ < t_d, [Q_1, Q_2], P_2 >,\ < t_d, [Q_2, C_2], P_3 >,\ < t_c, [C_2, Q_3], P_4 >\},\ < t_c, [Q_3, E], P_5 >\}$: the target object t_e is NN for the interval (subroute) $[S, Q_1]$, and the shortest path from the subroute to t_e is P_1; t_d is the NN for the subroute $[Q_1, Q_2]$ with the shortest path P_2; t_d is also NN for the subroute $[Q_2, C_2]$ with the shortest path P_3; t_c is that for the subroutes $[C_2, Q_3]$ and $[Q_3, E]$ with the shortest paths P_4 and P_5, respectively.

By proposing heuristics for selecting computation points (e.g., $\{S,\ C_1,\ C_2,\ C_3,\ C_4\}$ in the previous example) and initializing NN search region for these computation points, our CNN search finds the target objects with the shortest path length from all the points on the route effectively.

However, when this search is based on the real-time traffic conditions (e.g., the nearest target object is one with the shortest travel cost from current location), our method cannot be used directly because there are no perfect relations between the path length and the travel cost.

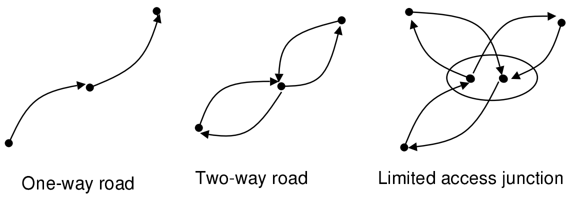


Figure 2: Representing different types of roads and junctions [6].

2.2 Integrated representation of traffic information and road network

To represent the traffic information on road network, a typical method [6] represents the road network using a directed graph. In the graph, each edge depicts a one-way road and each node corresponds to a junction. Two-ways roads can be presented as a pair of edges: one in each direction. This model permits easy modeling of one-way roads and limited access junctions. However, it keeps the topology relations among vertexes, and ignores the spatial relations among them. Extra nodes should be added to the graph when there are any access limitations (constraints of specific traffic controls). In other words, one cross node on the road network may be represented with several vertexes corresponding to the junctions, and they are independent with each other. Figure 2 gives the representation of different types of roads and junctions: one-way road, two-way road without any access limitations, and T-junction with some access limitations (the center point on this T-junction is represented by two vertexes in this directed graph).

Because this representation method ignores the spatial attributes of map objects, only the routing queries are applicable well on this model. For example, Lee's algorithm [7] is used for routing on this model. Lee's algorithm finds the best route with respect to optimizing some metric, as long as a route exists. The method simulates an inkblot spreading out over a piece of paper, centered on the start point. The covered area represents the already explored vertices. When the destination is reached, the algorithm traces the route back to the start.

Another method for representing the traffic cost is mentioned in [8]. They proposed an architecture for keeping traffic information on nodes of road network. However, the information of traffic constraints on the nodes is omitted in their discussion.

So we propose a method for representing traffic information including travel cost and traffic constraints on road network; especially we give a flexible representation method for traffic constraints. We also propose a heuristic method for temporal CNN search in taking the advantages of our represention method.

3 Integrated Representation Method

We propose a representation method for integrating traffic information and spatial information about road network by considering the followings:

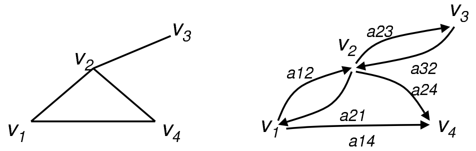
1) The traffic conditions change continuously, and the snapshot of conditions is recorded as traffic information. In comparing with the traffic information, the map of road network is seldom updated, and can be regarded as static information. Therefore, if the static information is managed by an efficient structure, the changes of traffic information associated with the road map should not disturb the stability of the structure.

2) The integrated representation should not only support the spatial query on road network and the temporal query on traffic information, but also support the interaction between these two kinds of queries.

3.1 Modeling of road network and traffic information

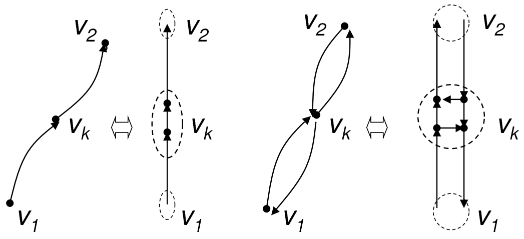
A road network with nodes and links representing the crosses and road segments can be regarded as a un-directed graph G, $G = (V, L)$, where V is a set of vertices $\{ v_1, v_2, ...v_n\}$, and L is a collection of lines $\{ l_1, l_2, ...l_m\}$. Traffic information on the road network is regarded as a directed graph G', $G' = (V, A)$, where V is a set of vertices $\{ v_1, v_2, ...v_n\}$, and A is a collection of arcs $\{ a_1, a_2, ...a_p\}$.

Figure 3 depicts these two kinds of graphs. In the un-directed graph of Figure 3 (a), road segments are represented by lines; while in the directed graph of Figure 3 (b), junctions are represented by arcs. One line for road segment in Figure 3 (a) may be corresponded to two arcs in Figure 3 (b) for two-direction traffic information. In addition to the directions of traffic, there are usually traffic controls (constraints) on road network: for example, the right-turn and U-turn are forbidden on some cross points, which constrain the action of traffic. The typical road junctions with (or without)



(a) (b)

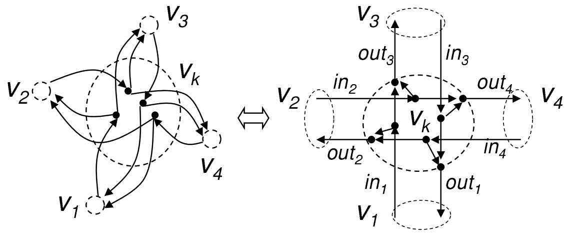
Figure 3: Road segment and traffic arc



(1) (2) (1) (2)

(a) (b)

Figure 4: One-way road and two-way road



(1) (2)

Figure 5: Cross node with constraint

constraints are given in Figure 4 and Figure 5. Road junctions are represented by using [6]'s model in (1) of two figures. Considering the shortcomings of this simple model, we propose a *super-node* representation method for integrating junctions (including traffic cost and traffic constraints) and road network.

A *super-node* can be defined as a node in road network with multiple corresponding junctions: for example, v_k in Figure 4 (a) (2), Figure 4 (b) (2) and Figure 5 (2). The information on the *super-node* contains the following parts (for simplicity of explanation, road junctions in Figure 5.(2) is used as an example):

1) *Cost-arc*: A cross node on road network, e.g. v_k in Figure 5(2), is a *super-node* for traffic information. The arcs which have v_k as their final vertex are called in-arcs, denoted as in_i, and similarly the arcs which have v_k as their initial vertex are called out-arcs, denoted as out_j. The number of those arcs is called as in-degree (e.g. 4) and out-degree (e.g. 4), respectively. Every out_i is defined as a *Cost-arc* consists of the final vertex of this arc and the traffic cost for traveling through this arc. *Cost-arc*s of v_k in Figure 5(2) are

$$\begin{bmatrix} out_1(v_1, cost_{k1}) \\ out_2(v_2, cost_{k2}) \\ out_3(v_3, cost_{k3}) \\ out_4(v_4, cost_{k4}) \end{bmatrix}.$$

2) *Constraint-matrix*: The constraints on the *super-node* can be represented with an $n \times m$ matrix *CM*:

$$CM(v_k) = \begin{array}{c} \\ in_1 \\ in_2 \\ \vdots \\ in_n \end{array} \!\!\begin{array}{c} \begin{array}{cccc} out_1 & out_2 & \dots & out_m \end{array} \\ \begin{pmatrix} C_{11} & C_{12} & \dots & C_{1m} \\ C_{21} & C_{22} & \dots & C_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ C_{n1} & C_{n2} & \dots & C_{nm} \end{pmatrix} \end{array}.$$

And

$$C_{ij} = \begin{cases} 1 & \text{there is restriction from } in_i \text{ to } out_j; \\ 0 & \text{there is a junction from } in_i \text{ to } out_j. \end{cases}$$

The *Constraint-matrix* for v_k in Figure 5.(2) is:

$$CM(v_k) = \begin{array}{c} \\ in_1 \\ in_2 \\ in_3 \\ in_4 \end{array} \!\!\begin{array}{c} \begin{array}{cccc} out_1 & out_2 & out_3 & out_4 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \end{array}.$$

where there are restrictions on going from in_1 to out_1 and out_4, from in_2 to out_1 and out_2, from in_3

to out_2 and out_2, and from in_4 to out_3 and out_4. If there is no restriction for any in_i of the super-node v_k, the *Constraint-matrix* of v_k is filled with 0, and is regarded as $\emptyset$.

This method decreases the redundancies in the database by adopting a complex node representation. Though the total information for the network may be more than those in [6]'s method, it is easy to integrate the traffic information and the static road network, and this method does not injure the stability of the spatial index structure for road network.

4 Temporal Continuous Nearest Neighbor Search

In this section, we propose a method for CNN search along a predefined route based on a dataset, denoted as *super-node* dataset, which is generated by the *super-node* representation method proposed in the previous section. The predefined route from a start point v_1 to an end point v_n is given by an array $Route(v_1, v_n) = (v_1, v_2, ..., v_{n-1}, v_n)$, and the target object set $\{t_a,\ t_b,$ $...\}$ is managed by a spatial index structure (e.g. R-tree [9]). The detailed discussions about the processing of target objects by taking advantages of spatial index structure can be found in our previous papers [5, 10]. We center on the *super-node* repressentation method and its influence on CNN search. The *super-node* dataset consists of information about road network and traffic cost on the network. To simplify the explanation, we first use an abstract *cost* on road network, and in the next section analyse the concrete examples of *cost*.

4.1 Observation of super-node dataset in CNN search

We make observations of the *super-node* dataset in the CNN search process:

1) Every vertex in the *super-node* dataset keeps the cost information of the possible out-arcs, so the cost of traveling from a vertex v_i on $Route(v_1, v_n)$ to the following vertex v_{i+1} along this route is kept on vertex v_i and denoted as $v_i.cost_{i+1}$. If the nearest neighbor (NN) of v_{i+1} is known as t_{i+1} with $cost(v_{i+1}, t_{i+1})$, the cost of traveling from v_i to its NN t_i is not larger than a value *Cost-limit* (v_i), which is computed by:

$$\textit{Cost-limit}\ (v_i) = v_i.cost_{i+1} + cost(v_{i+1}, t_{i+1})$$

Cost-limit (v_i) is used to set a region for the NN search of v_i (e.g. in Figure 6), the NN of v_i can only be found inside the dotted circle region. The region is defined as a circle with radius of *Cost-limit* (v_i) and center of v_i.

2) The nearest target object t_{i+1} of v_{i+1} is also the nearest one on the possible paths from v_i via v_{i+1}. In other words, t_{i+1} is the nearest one found on a path from v_i via $v_i.out_{i+1}$. If there is any object being nearer to v_i than t_{i+1}, the shortest path from v_i to it does not pass by v_{i+1}. Certainly, it is possible that there is a path from v_i to t_{i+1} via v_j $(j \neq i+1)$, which is shorter than *Cost-limit* (v_i). This situation is depicted in Figure 6, where v_{i-1} and v_{i+1} share the same NN t_{i+1}, but there is no overlap between the two paths p_{i+1} and p_{i-1}.

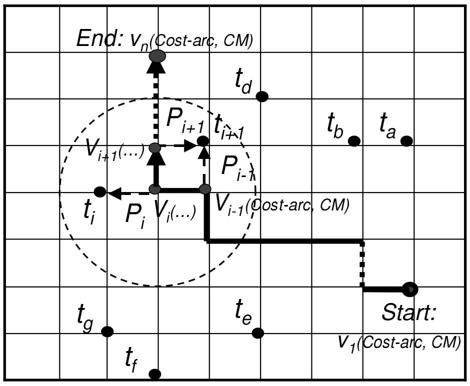


Figure 6: Predefined route and NN for v_i.

Based on the previous observations, it can be concluded that:

1) The path length from v_i to the nearest neighbor (NN) t_{i+1} of v_{i+1} can be set as a limit for the NN search of v_i.

2) The NN search of v_i can be executed along the out-arcs of v_i except $v_i.out_{i+1}$.

Here, we give a simple proof for these conclusions:

1) We prove that t_{i+1} is a candidate of NN search of v_i. Based on the definition of $Route(v_1, v_n)$, v_i and v_{i+1} are along the same route, so there is an out-arc of v_i leading to v_{i+1}. Being the NN object of v_{i+1}, t_{i+1} can also be reached from v_i via v_{i+1}. Therefore, t_{i+1} is possible to be the NN of v_i, too.

2) We prove that *Cost-limit* (v_i) is the shortest one from v_i to any object via v_{i+1}. If there is another object t' with a path shorter than that of v_{i+1} via v_{i+1}, then t' is also nearer to v_{i+1} than t_{i+1}. This contradicts to the promise that t_{i+1} is the NN of v_{i+1}.

4.2 Reverse search method of CNN

We propose a method for CNN search along $Route(v_1, v_n)$. This method begins from the end vertex of this route, and searches the NN for every vertex in the reverse order of this route.

Our method first searches t_n for the end vertex v_n; and then generates a search limit for the next computation vertex v_{n-1} based on the previous result, and checks whether there is an object nearer to v_{n-1} via the out-arcs of v_{n-1} except $v_{n-1}.out_n$. These steps run in cycle until the computation vertex is v_1. This method is correct, based on the previous observations.

The NN search for every vertex can be realized by adopting a priority queue to maintain the current frontier of the search. Any vertex with a higher cost from v_i than the limit value is not insert into the queue. By expanding the vertex on the head of the queue, the algorithm ends when the head vertex connects to a target object.

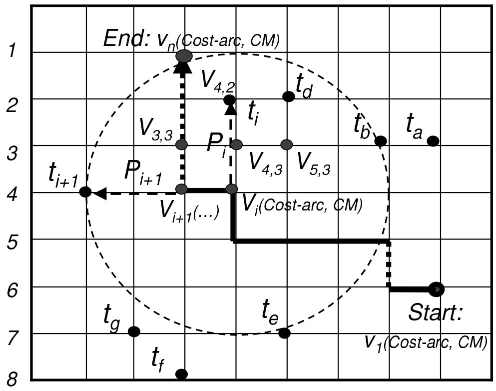


Figure 7: NN search for v_i with a limit.

An example for NN search of v_i is given in Figure 7. The NN of v_i is to be searched inside the dotted region. There are assumptions that every grid represents a unit of cost; right-turn and U-turn are forbidden on v_i; and no restrictions are imposed on vertex $v_{4,3}$. The search for v_i begins from the following possible out-arcs of v_i: here, $v_i.out_{4,3}$. As there is no target object connecting to $v_{4,3}$, and the cost from v_i to $v_{4,3}$ is not larger than *Cost-limit* (v_i), the search expands the vertex $v_{4,3}$ using the width-first method. The vertex $v_{4,2}$ connecting to a target object t_i with the lowest cost is found, and the object t_i is regarded as the NN of v_i. The main steps in this algorithm are given here:

==========================

```
Algorithm NN-search
/* Input: $Route(v_1, v_n)$, target object set T, source point $v_i$,
      and $< t_{i+1}, interval_{i+1}, path_{i+1}, cost(v_{i+1}, t_{i+1}) >$;
   Output: a triple $< target, interval, path, cost >$*/
1. Initialize priority queue $Queue$;
2. $Cost\text{-}limit\ (v_i) = v_i.cost_{i+1} + cost(v_{i+1}, t_{i+1})$;
3. Sort $v_i$ with $(cost = 0)$ and
      $t_{i+1}$ with $(cost = Cost\text{-}limit\ (v_i))$ into $Queue$;
4. Do steps 5 to 6 unless $Queue$ is Null;
5. If the head of $Queue$ is $v_k$ not connecting to
      a target object
   Then
   (1)  Expand possible out-arcs of $v_k$: $v_k.out_p$,
        where $C_{kp}$ <> 1 in $CM(v_k)$ ;
   (2)  Compute the cost from $v_i$ up to $v_p$;
   (3)  If the cost is smaller than $Cost\text{-}limit\ (v_i)$
        Then sort $v_p$ into $Queue$;
6. If the head of $Queue$ is a node connecting to
     a target object $t_i$
   Then return result of triple
   $< t_i, interval_i, Path_i, cost(v_i, t_i) >$
```

==========================

5 Analysis and Conclusion

5.1 Analysis

The anlysis is done from a view point of providing concrete examples of *cost*: when there is a uniform speed of traffic on the road network, the *cost* can be the length of the road segment; otherwise, the *cost* can be the travel time for every traffic arc on the road network. Certainly, there are other kinds of *cost*: for example, the toll of a path. Our method supports the search based on all these *cost* definitions.

The discussion and examples used in the previous sections can be regarded as the traffic arcs with the assumption that the *cost* is equal to the length of the road segment, implicitly. If the *cost* is the travel time on the traffic arc, though the region may be not a circle on the road map, it is sure a region can be generated with the same method, and the NN search for every vertex can be executed using the same program. This is because the region is actually realized by adopting the priority queue ordering on *cost*.

Moreover, for some important route planning problems, other costs may be identified: namely, turn costs appear when we make a turn on a cross point. It also can be regarded as the cost of leaving one traffic arc and entering the next [11] via their shared node. Our method is able to process the turn cost by extending the *Constraint-matrix* to a *Turn-Cost/Constraint-matrix*. The *CM* defined in Section 3 can be modified to a *Turn-Cost/Constraint-matrix*:

$$T\_CM(v_k) = \begin{array}{c} \\ in_1 \\ in_2 \\ \vdots \\ in_n \end{array} \begin{array}{c} \begin{array}{cccc} out_1 & out_2 & \dots & out_m \end{array} \\ \begin{pmatrix} TC_{11} & TC_{12} & \dots & TC_{1m} \\ TC_{21} & TC_{22} & \dots & TC_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ TC_{n1} & TC_{n2} & \dots & TC_{nm} \end{pmatrix} \end{array}$$

And

$$\begin{cases} 0 \le TC_{ij} < Max & \text{the turn cost from } in_i \text{ to } out_j; \\ TC_{ij} = Max & \text{there is restriction from } in_i \text{ to } out_j. \end{cases}$$

For example, the *Turn-Cost/Constraint-matrix* for v_k in Figure 5.(2) may be like this:

$$T\_CM(v_k) = \begin{array}{c} \\ in_1 \\ in_2 \\ in_3 \\ in_4 \end{array} \begin{array}{c} \begin{array}{cccc} out_1 & out_2 & out_3 & out_4 \end{array} \\ \begin{pmatrix} MAX & 10 & 40 & MAX \\ MAX & MAX & 10 & 30 \\ 10 & MAX & MAX & 30 \\ 10 & 40 & MAX & MAX \end{pmatrix} \end{array}.$$

where the MAX is defined as a large constant value. The element TC_{ij} in this matrix with a value of MAX represents a restriction from in_i to out_j: e.g. U-turn and right-turn are forbidden in this example, so MAX is assigned to TC_{ii} ($i = 1, 2, 3, 4$), TC_{14}, TC_{21}, TC_{32} and TC_{43}. The value of TC_{12} represents the cost 10 (e.g. 10 seconds) of making a left-turn on the cross point (from in_1 to out_2), while the cost of crossing the point v_k from in_1 to out_3 is 40.

The values of the turn cost can be used naturally in the process of searching algorithm proposed in this paper.

On the other hand, either kind of cost is adopted in the dataset, and the quantity of information on every vertex keeps the same. Therefore, when the road map is managed by some spatial index (e.g. R-tree), *Cost-arc* and *Constraint-matrix* associated to a vertex are stored into a fixed space of specific diskpage. The update for traffic information will not injure the stability of the spatial index.

5.2 Conclusion

In this paper, we proposed a representation method for integrating traffic cost and spatial road network, and a method for CNN search based on this representation. Our method is flexible for representing *cost* on traffic network, and the reverse search method for CNN taking the advantages of this representation method can be realized efficiently. The evaluation of our method with a prototype system is in our future work.

References

[1] S. Shekhar and D.R. Liu: "CCAM: A Connectivity-Clustered Access Method for Aggregate Queriees on Transportation Networks: A Summary of Results", *Proc. of ICDE'95*, pp. 410–419 (1995).

[2] Y.F. Tao, D. Papadias and Q.M. Shen: "Continuous Nearest Neighbor Search", *Proc. of VLDB'02*, pp. 287–298 (2002).

[3] Z.X. Song and N. Roussopoulos: "K-Nearest Neighbor Search for Moving Query Point", *Proc. of SSTD'01*, pp. 79–96 (2001).

[4] S. Bespamyatnikh and J. Snoeyink: "Queries with Segments in Voronoi Diagrams", *SODA* (1999).

[5] J. Feng and T. Watanabe: "A Fast Method for Continuous Nearest Target Objects Query on Road Network", *Proc. of VSMM'02*, pp. 182–191 (2002).

[6] J.Fawcett and P.Robinson: "Adaptive Routing for Road Traffic", *IEEE Computer Graphics and Applications*, Vol. 20, No. 3, pp. 46–53 (2000).

[7] C.Y. Lee: "An Algorithm for Path Connectivity and its Applications", *IRE Trans. on Electronic Computers*, Vol. 10, No. 3, pp. 346–365 (1961).

[8] D. Papadias, J. Zhang, N. Mamoulis and Y.F. Tao: "Query Processing in Spatial Network Databases", *Proc. of VLDB 2003*, p. to appear (2003).

[9] A. Guttman: "R-Trees: A Dynamic Index Structure for Spatial Searching", *Proc. of ACM SIGMOD'84*, pp. 47–57 (1984).

[10] J. Feng and T. Watanabe: "A Fast Search Method of Nearest Target Object in Road Networks", *Journal of the ISCIE, to appear*, Vol. 16, No. 9 (2003).

[11] S. Winter: "Modeling Costs of Turns in Route Planning", *GeoInformatica*, No. 4, pp. 345–361 (2002).