International Journal of Geo-Information

MDPI

Article

A Novel Dynamic Physical Storage Model for Vehicle Navigation Maps

Shaohua Wang [1,2], Ershun Zhong [1,*], Kai Li [2], Guanfu Song [2] and Wenwen Cai [2,*]

[1] Institute of Geographic Sciences and Natural Resources Research, Chinese Academy of Science, Beijing 100101, China; wangshaohua@lreis.ac.cn

[2] SuperMap Software Co. Ltd., Beijing 100015, China; leek13@163.com (K.L.); songguanfu@supermap.com (G.S.)

* Correspondence: zhonges@supermap.com (E.Z.); caiwenwen_bnu@163.com (W.C.); Tel.: +86-10-5989-6655 (E.Z.); +86-10-5989-6521 (W.C.)

Academic Editors: Georg Gartner, Haosheng Huang and Wolfgang Kainz
Received: 20 February 2016; Accepted: 18 April 2016; Published: 22 April 2016

Abstract: The physical storage model is one of the key technologies for vehicle navigation maps used in a navigation system. However, the performance of most traditional storage models is limited in dynamic navigation due to the static storage format they use. In this paper, we proposed a new physical storage model, China Navigation Data Format (CNDF), which helped access and update the navigation data. The CNDF model used the reach-based hierarchy method to build a road hierarchal network, which enhanced the efficiency of data compression. It also adopted the Linear Link Coding method, in which the start position was combined with the end position as the identification code for multi-level links, and each link traced up-level links consistently without recording the array of identifications. The navigation map of East China (including Beijing, Tianjin, Shandong, Hebei, and Jiangsu) at 1:10,000, generated using the CNDF model, and the real time traffic information in Beijing were combined to test the performance of a navigation system using an embedded navigation device. Results showed that it cost less than 1 second each time to refresh the navigation map, and the accuracy of the hierarchal shortest-path algorithm was 99.9%. Our work implied that the CNDF model is efficient in vehicle navigation applications.

Keywords: dynamic navigation maps; physical storage format; reach-based hierarchical network; linear link coding

1. Introduction

The physical storage format (PSF) of navigation maps is one of the key technologies for vehicle navigation systems. It aims to reconstruct the structure of spatial data to satisfy the requirements of vehicle navigation. Some large companies and organizations have proposed their own PSF used for navigation products (Table 1).

Table 1. Major Physical Storage Format (PSF) used for Vehicle Navigation Maps.

PSF Name	Owner	Information
Car IN	Siemensvdo	http://www.siemensvdo.com
KIWI	KIWI forum	http://www.jsa.or.jp
SDAL	NAVTEQ	http://www.navteq.com
Siemens AF	Siemensvdo	http://www.siemensvdo.com
TRAVELPILOT	blaupunkt	http://www.blaupunkt.com
PSI	BMW	http://www.psf-initiative.com

When the PSF format was initially proposed, the major concern for onboard devices was how to quickly access navigation data due to poor device performance. Therefore, a large number of static PSF models were developed to facilitate data access. Static models were used for read-only storage media, among which the SDAL and KIWI were the representative models at that time.

SDAL is a PSF model developed by the NAVTEQ Corporation [1]. It uses the global KD-tree index to manage spatial data and topology data in a multi-scale structure. The spatial data and topology data are stored by parcels according to the spatial bounding rectangle of KD-tree nodes. The parcel is the basic element for data access. Therefore, it is difficult to control the memory overhead in route analysis, since the search radius cannot be recorded due to the storage style of the spatial and topology data used by SDAL. Besides, SDAL uses physical address to associate all data, which indicates that it does not support data updates.

KIWI is another popular PSF model used for vehicle navigation [2]. It was first developed by the Japanese KIWI forum and then became a published industry standard in Japan. The basic element of KIWI is the data frame. All spatial data frames are clustered by varisized grids defined by a hierarchical grid index. The spatial data and topology data in KIWI are partitioned by Region, a preprocessed sub-graph satisfying the complete connectivity for search space controlling. Therefore, KIWI has resolved the problem for search bound controlling faced by SDAL. However, the hierarchy in KIWI is determined based on street types, which requires manual tuning of the data to trade-off between computing speed and sub optimality of the computed routes. In addition to SDAL and KIWI, ISO TC204/WG3 has proposed the PSF world standard [3].

The static PSF models imply two major shortcomings that hinder their applications in dynamic vehicle navigation. First, their storage models are designed for read-only access mode, which does not allow for data updates. Second, their hierarchical methods are based on natural road classes, which cannot maintain the consistency of road networks with mathematic theorems. To resolve the problems, as well as maintain the successful ratio of route planning output, several artificial processions need to be invoked during the compilation process. However, the compilation software for the static PSF models is very complex and fails to be adapted by the dynamic navigation applications.

Therefore, it is necessary to develop dynamic PSF models for navigation maps. To design high performance dynamic PSF models, some issues must be taken into account. For example, the large quantity of spatial data is contradictory with the limited disk storage of the navigation devices. The requirements of real-time response for intensive redrawing of the navigation maps is constrained by the poor memory capacity of the navigation equipment. Besides, the high complexity of the shortest-path analysis algorithms may be ineffective, considering the weak floating-point operations capacity and limited computing speed of the navigation facilities.

The development of communication technology has promoted the generation of dynamic PSF models. The key research content for developing dynamic PSF models is to implement the update methods for local navigation data. The update methods can be divided into replacement update methods or incremental update methods. For replacement update methods, such as the Japanese i-Format, the local navigation data are updated each time by downloading the new map version, since the memory cache of mobile or embedded clients is supposed to be small. Therefore, it will result in large traffic costs when using replacement update methods.

For incremental update methods, the update of local navigation data each time focused on updating and modifying the records that needed to be transferred and appended to the data. However, it is challenging to modify the high coupling local data and guarantee the high speed of data access simultaneously. Researchers have focused on improving the basic methods and have proposed several applied models [4–8]. For example, Xu modeled a self-organizing, distributed traffic information system built upon vehicle-to-vehicle information exchange for route optimization [4]. The Linear Reference System and Dynamic Segment (LRS&DS) is considered as the primary model for managing dynamic traffic information, such as real-time traffic information, which is indispensable for dynamic navigation.

In this paper, we developed a novel PSF model, *i.e.*, the China Navigation Data Format (CNDF), for spatial data used by dynamic navigation. It uses the linear link coding method, which is effective in saving disk storage for updating real-time traffic information and supports fast hierarchical records mapping. The network model adapted from the LRS&DS algorithm was used by CNDF to code the link of the hierarchical reach-based network model. Our model has the capability of accelerating route planning by using a hierarchical network. It is also able to rapidly map target records to each level of the road network and readjust related records to ensure hierarchical consistency of the model after data updating. Therefore, the dynamic PSF model, CNDF, can be used for incremental updates.

The CNDF has passed the review of the National Standardization Management Committee, and will be officially published as a national standard. The national standard is compiled with the existing standard of ISO 20452, but it also considers specific navigation conditions in China, including the huge amount of data, the fast speed of road construction, the transport rules and regulations, the habits of location names, and the specification of basic data. In Section 2, we illustrate its structure and implementation. Section 3 shows an experiment and the corresponding results using CNDF for navigation maps of East China on a navigation device. Section 4 contains the conclusion. Our work indicated that the CNDF model is efficient for dynamic navigation.

2. Methodology

In this section, we illustrate the key technologies used for developing the CNDF model, including the physical structure of the spatial data, computing model for route analysis, and the data update mechanism.

2.1. The Structure of the CNDF Model

2.1.1. Multi-Scale Grid Index

Spatial data using the CNDF model are clustered by multi-scale grids. All of the grids are indexed by the multi-scale grid (MSG) index. The MSG is made up of groups of level grid indexes (LG) and each LG manages data at a specific scale (Figure 1). LG is a variant Quad-tree [9]. The grids of LG are divided into four varisized ranks, including a general grid (GG), middle grid (MG), detailed grid (DG), and extended grid (XG). The width (or height) of the grid within the same rank is equivalent. DG is the leaf nodes of LG, and spatial data at a specific scale are clustered into pages by DG. In some places, the data size may overflow, for example, the CBD in large cities. In this case, MSG allows DG to be further partitioned into smaller XGs. For spatial queries, the addresses of target pages can be found by searching the path from GG to DG (even to XG) that cross by query bounds.

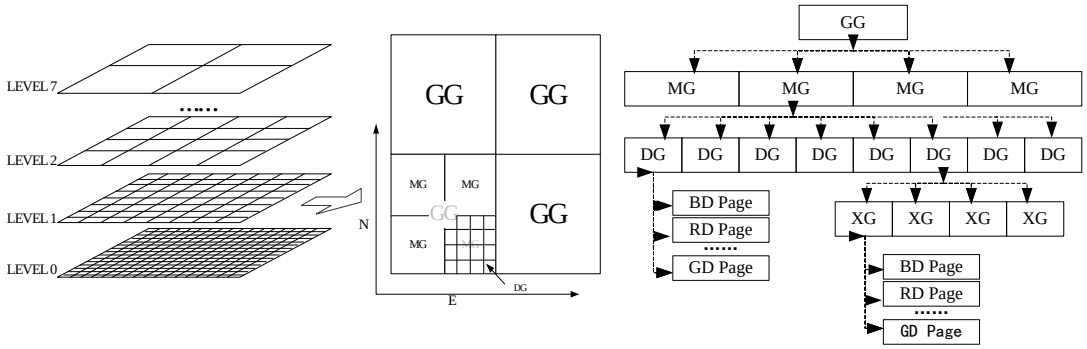


Figure 1. Illustration of the Multi-Scale Grid index.

MSG is suitable for a navigation system for three reasons. First, MSG is compact, which does not need to ensure a minimum rectangle boundary (MRB) in leaf nodes (DG or XG). The grid width (or height) of each rank can be calculated by the reference grid (the lowest level DG) using Equation (1).

Secondly, MSG is a high performance index. It regulates β denoted by $2n$ ($n \geqslant 0$), which avoids the float point operation for a spatial query. Further, MSG is close to being balanced. Finally, MSG has a large capacity index. Suppose M represent the capacity of a single LG, where M = $\alpha^2 \times \beta^2 \times \gamma^2 \times$ max page size (MPS). If MPS = 64 KB, and $\alpha = \beta = \gamma = 16$, then M = 1 TB. Therefore, the capacity of MSG is sufficient to manage spatial data covering vast areas.

$$w_i = w_0 \prod_{k=1}^{i} \beta_k,\ h_i = h_0 \prod_{k=1}^{i} \beta_i,\ i > 0 \tag{1}$$

where w and h represent the width and height of the grid, respectively, and β represents the ratio between two adjacent ranks.

2.1.2. Logical Structure of the CNDF Model

Figure 2 illustrates the logical structure of the CNDF model. The spatial data using the CNDF consist of multi-level background data (DB), multi-level road data for display purposes (RD), multi-level road data for route planning (RP), driver guidance data (GD), geocoding data (GC), and land mark data (POI). The management data include the volume manager data (MD), spatial index (SIDX), and attribute index (AIDX).

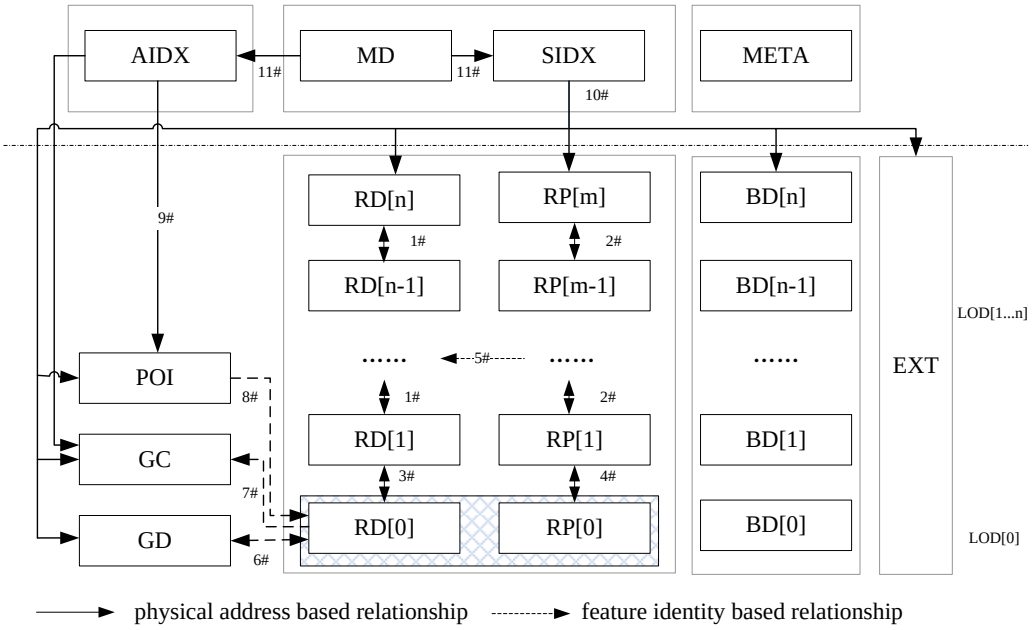


Figure 2. Logical structure of the China Navigation Data Format (CNDF).

All the spatial data are organized and indexed by the MSG index according to their scales. Each leaf node of MSG points to a group of pages defined by a rectangle. Each link of the road network has a unique identity code comprised of the start node tag and the end node tag assigned by the linear link coding (LLC) method. The LLC method is illustrated in Appendix A. In this way, a connotative mapping relationship is built on links in multi-levels representing a road segment.

Applications can trace the records stored in different places based on the hierarchical relationship. LLC does not record the volatile physical address associated with the data, but stores the relationship. Storing a physical address is difficult to maintain the correct topology relationship of the data, since data modification or updates may result in obstacles [10]. The hierarchical mapping relationship between levels is implicated in the LLC code. Compared with the method of storing relationships by recording all identity number of links between upper and lower levels, the CNDF can reduce data redundancy and save disk storage by using LLC.

The data structure of the CNDF is efficient and flexible through adopting the reach based strategy, since the reach metric is heuristic knowledge derived from a mathematical theorem rather than an empirical value. The range of readjustment is evaluated in Section 2.2.3, which proves the time cost for link updates in CNDF is convergent. Further, the data consistency after the link update can be maintained.

2.1.3. Physical Structure of the CNDF Model

Figure 3 demonstrates the physical structure of the CNDF model, including file, page, and block. The file contains the pages. Spatial data and non-spatial data are clustered into pages with different strategies. Spatial data are clustered using varisized grids defined by MSG. Non-spatial data are portioned into pages according to the order of their prime key.

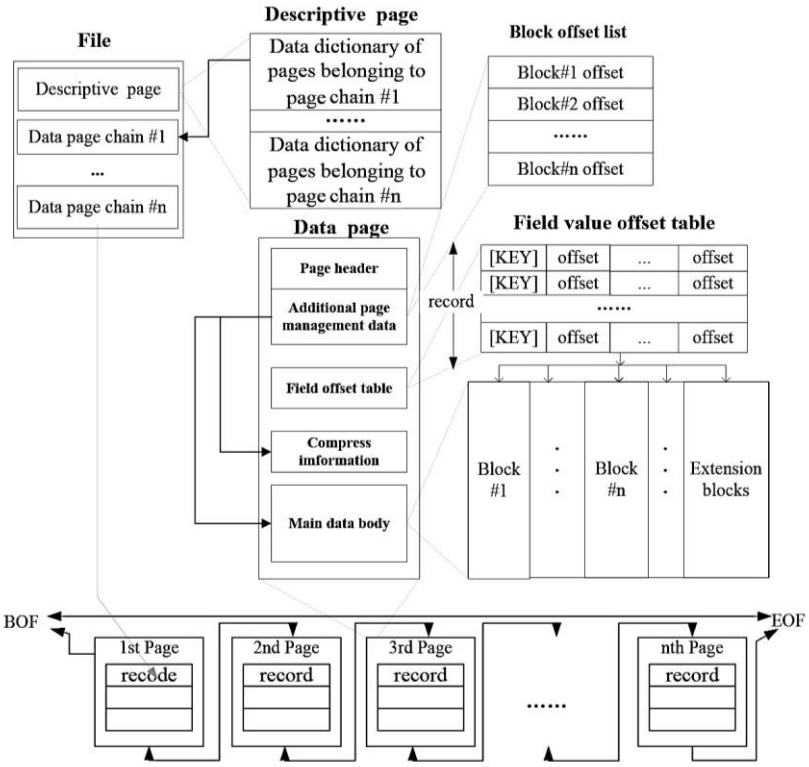


Figure 3. Physical structure of the China Navigation Data Format (CNDF).

Pages are comprised of a page header and several blocks. The page header records necessary information pertaining to the page structure, including the compress flag, block number, and block offset from the beginning of the page. The page header also assigns two address pointers, used to save the previous page and the resultant page. In this way, all pages form a chain, optimizing page insertion under the restriction of maintaining the record order. The first block of the page is the offset table, used to store the offset of each field. The table is updated correspondingly when the field is lengthened. Records in the CNDF are in row order, which is different from other PSFs. Records in row order can avoid the reading of unnecessary fields into the memory. In CNDF, the page size is limited within MPS to 128 KB.

2.2. Route Computing Models of the CNDF Model

Many Studies have focused on the optimal route analysis algorithms suitable for vehicle navigation and most of them rely on a specific topological structure of the road network [11–14]. For example, the natural road hierarchy network is a kind of topological structure used by hierarchical

network models and is popular for navigation applications. However, it requires manual tuning of the data and its performance depends on a delicate trade-off between computing speed and sub optimality of the computed routes. Since the referred classes of roads are artificially defined according to experience, the route search based on natural road hierarchy network can only yield proximate solutions.

The natural road hierarchy networks is also not suitable for dynamic data update. The route search may fail if the data are modified or not maintained. The maintenance operations are therefore essential for the hierarchical network models using the natural road hierarchy networks. The target of the maintenance operations is to maintain the constraints of the road networks predefined by the models, such as connectivity and searching bounds, since the constraints may be changed or even be invalid if the road network is modified. In addition, there is no mathematic method used to evaluate the performance of the maintenance operations, and the cost, which is unpredictable and expensive, becomes a tremendous obstacle for dynamic data updates.

Therefore, we constructed the reach-based hierarchical network, inspired by the reach-based heuristic route search algorithm. In the following section, we define the reach-based hierarchical network and propose a shortest-path analysis algorithm based on the network, and also discuss the scope evaluation of the readjustment.

2.2.1. The Reach-Based Hierarchical Network

The mathematic definition of reach was given by [15,16]. By computing the reach value of each link of the input road network, the reach values can be partitioned into sequence groups using Equation (2). The road network in the *i*th level is reconstructed by links where $r(e) \geqslant r^i_{min}$, and it is simpler than that in the (*i-1*)th level. Part of the records in a level are used to save the relationship with the corresponding records in its upper level. In this way, multi-level topology is built up.

$$RL = \left\{ R_i \mid R_i = \left\{ r \mid r^i_{min} \leqslant r\,(e) \ \leqslant \ r^i_{max} \right\} \right\},\ i = 0,\ 1,\ \ldots,\ n-1 \tag{2}$$

where *RL* represents the sequence groups-defined levels; R_i represents the group of the reach value in level *i*; r^i_{min} and r^i_{max} represent the minimum and maximum reach values, respectively, in level *i*; and *r(e)* represents the reach value of link *e*.

The route search using the reach-based hierarchical model is more effective than the single-level model-assigned reach values as hieratic information, since the number of links decreases sharply with increasing levels. It is very useful for navigation devices with low speed CPU and limited memory. The designation of *RL* is important for model optimization. The redundancy of the navigation map is directly proportional to the total number of levels. However, the convergence speed of the algorithm is diversely proportional to the difference between r^i_{min} and r^i_{max}.

2.2.2. The Reach-Based Hierarchical Shortest-Path Algorithm

Searching in a single level in the reach-based hierarchical shortest-path (RHSP) algorithm is similar to that in Dijkstra's algorithm. In Dijkstra's algorithm, the shortest-tree is grown to contain the shortest paths from the start node to all other nodes. During the growing process, each node is marked with "Unfound", "Found", or "OnPath". Table 2 shows the two phases of the multi-level search, *i.e.*, the upward search and the downward search. The upward search is the first phase, which can be implemented according to pseudo code shown in line (3) to line (15) in Table 2. In the upward search, the reach bounds increase iteratively, so all the nodes with $b_i \geqslant r(v) \geqslant m(s, v, P)$[1] can be guaranteed to be marked in the *i*th level of the road network, in the worst situation. The downward search is the second phase, which can be implemented according to the pseudo code shown in line (16) to (26) in Table 2. In the downward search, the reach bounds decrease iteratively; all nodes with $b_i \geqslant r(v) \geqslant d(v, t)$[2] can be guaranteed to be marked in the *i*th level of the road network, in the best situation.

Table 2. The Pseudo Code of the Reach-Based Hierarchical Shortest-Path Algorithm.

FUNCTION RHSP
/*SS[level][node] is the set of start nodes of each level*/
/*TT[level][node] is the set of end nodes of each level*/
/*HEAP is sorted by m(s, v), implemented according tot he binary heap or Fibonacci heap*/
/*RB[level] is the bound array of each level{b_0, b_1, b_2, .., b_n};*/
/*RHTopo is the reach-based hierarchical network*/
FUNCTION RHSP(s ,t ,RHTopo, RB[])
(1)COST = 0; LEVEL = 0;
(2)Union(SS[0], s); Union(TT[0], t); Insert(HEAP, s, COST);//Init start set, end set and heap
(3)WHILE ((u = ExtraMin(HEAP, COST) ! = NULL)//Search higher level Node
(4) IF Within (SS[0], u) THEN RETURN COST;//target is found.
(5) IF Reach(u) < RB[LEVEL] THEN CONTINUE;//Filter lower lev-el node last loop left
(6) IF COST>RB[LEVEL+1] AND LEVEL+1 $\leqslant$ MAXLEVEL THEN
//Search upper-level nodes around target node, whose distance to t within bi at end side
(7) ENDUP = QuerybyPoint(RHTopo, point(x(t), y(t)), RB[LEVEL+1]));
(8) Union(TT[LEVEL +1], ENDUP); //Record the targets of upper level
(9) LEVEL++;//Begin higher level search
(10) CONTINUE;
END IF
(11) IF Reach(u) > RB [LEVEL] THEN Union(SS[LEVEL+1], u);//Save higher level node
(12) FOR EACH v IN Adjoin(RHTopo, u);//Get each node adjoin to u for spread
(13) IF OnPath (HEAP, v) THEN CONTINUE;
(14) IF UnFound (HEAP, v) THEN Insert(HEAP, v, COST + m(u, v));
(15) ELSE IF COST+ m(u, v) < m(HEAP, s, v) THEN Update(HEAP, v, COST + m(u, v));
NEXT v
NEXT
(16) FOR LEVEL= MAXLEVEL -1 TO 0//Search lower level node iteratively until target t is found
(17) FOR EACH u IN TT[LEVEL]//Get target nodes of higher level as start node of lower level
(18) IF !OnPath(HEAP, u) THEN CONTINUE;//filter the candidate targets not on path
(19) Insert(HEAP, u, m(HEAP, s, u));//Re-insert to heap with path cost m(s, u, P)
NEXT u
(20) WHILE ((u = ExtraMin(HEAP, COST) != NULL)//Search Lower level Node
(21) IF Within (SS[0], u) THEN RETURN COST;//target is found.
//control the search space is not over bi+1
(22) IF Distance((x(t), y(t), x(v), y(v)) > RB[LEVEL+1] THEN CONTINUE;
(23) FOR EACH v IN Adjoin(RHTopo, u);//Get each node adjoin to u
(24) IF OnPath (HEAP, v) THEN CONTINUE;
(25) IF UnFound (HEAP, v) THEN Insert(HEAP, v, COST + m(u, v));
(26) ELSE IF COST+ m(u, v) < m(HEAP, s, v) THEN
Update(HEAP, v, COST + m(u, v));
NEXT v
NEXT u
NEXT LEVEL
(27)RETURN -1; //Not found target

[1] $m(s, v, P)$ represents cost from start node (s) to current node (v) on current search path (P); [2] $d(v, t)$ represents the distance between current node (v) and terminal node (t).

We can deduce the properties that result in optimal RHSP.

All nodes on P must be marked "OnPath" by RHSP. Given the node list $V < v_1, v_2, v_3 \ldots v_n >$ on P, V can be divided into two subsets: $V_1 = \{u \mid r(u, P) = m(s, u, P)\}$ and $V_2 = \{v \mid r(v, P) = m(v, t, P)\}$. In V_1, for $r(u, P) \leqslant r(u)$, we have $m(s, u, P) \leqslant r(u)$. Similarly in V_2, for $d(v, t) \leqslant m(v, t, P)$, we have $d(v, t) \leqslant r(v)$. Therefore, the nodes of V_1 can be marked in the upward search and the nodes of V_2 can be recorded in TT first and then marked in the downward search.

All nodes that are not marked by RHSP must be not on path P. Suppose that node w on P is not marked by RHSP, so that $m(s, w, P) > r(w)$ and $d(w, t) > r(w)$. Since $r(w) < Min\ \{m(s, w, P), d(s, w, t)\} \leqslant \{m(s, w, P), m(w, t, P)\} = r(w, P)$, which conflicts with $r(w) \geqslant r(w, P)$, so w does not exist. Therefore, the result of RHSP is optimal. Compared to searching in the bottom level, the RHSP algorithm reduces the "ExtraMin" time in searching in the higher level, since nodes with little reach are abstracted in the bottom level.

In addition, the memory cost of RHSP can also be evaluated. Suppose that M_i is number of the nodes located in the cycle with radius equal to the reach bound of level i, and N is the number of nodes in the upper level of level i, so the node number of the upper bound inserted to Heap (*i.e.*, $O(C)$) can be calculated by Equation (3).

$$\mathrm{O}(C) \leqslant \sum_{i=0}^{n-1} 2M_i + N \tag{3}$$

2.2.3. The Scope Evaluation of Readjustment

The route distance is considered as the classification parameter in the reach-based hierarchical network. The road network is assumed as a planar graph. Given link (u, v) with $m(u, v) \geqslant d(u, v)$, where $d(u, v)$ is the Euclidean distance between two nodes and m represents the route length from link u to link v, we can deduce the property below.

The $e(u, v)$ and $e'(u', v')$ exist and satisfy $r(e) < r^i_{max}$ and $d(e', e) > r^i_{max}$. If the cost of e' changes and $r(e)$ remains unchanged, then $d(e', e)$ is the minimum value in $\{d(u, v'), d(u', v), d(u', u), d(v, v')\}$.

According to the definition of reach, given a directed graph $G = (V, E)$ with positive weights, path P of G starting at node s and ending at node t, and node v on path P, we can infer that the reach of v on P (*i.e.*, $r(v, P)$) is the minimum of $\{m(s, v, P), m(v, t, P)\}$, and the reach of v in G (*i.e.*, $r(v, G)$) is the maximum of $r(v, Q)$, where Q is set of least-cost paths cross node v, and the reach of link e (u, v) is minimum of $\{r(u), r(v)\}$.

If $e' \notin Q(e)$ after the cost of e' is changed, then the reach of e is not affected by the cost of e';

If $e' \in Q(e)$ after the cost of e' is changed, it must be the maximum cost path $P(s, t)$ of $Q(e)$ so that $e', e \in P$, assume that $m(v, t) = m(v, u', P) + m(u', t, P)$. Because $m(v, t) \geqslant d(e', e) > r^i_{max}$, $m(v, t) \geqslant r^i_{max}$. Then, as $r(e) < r^i_{max}$, $r(e) = r(u) = m(s, u) < m(u, t)$. Even if the cost of e' has changed, it still satisfies $m(v, t) \geqslant d(e', e) > r^i_{max}$. Therefore, $r(e)$ equals $r(u)$ is still unchanged. Assume that $m(u, t) = m(s, v', P) + m(v', u, P)$; the proof is the same as the above description. Therefore, the cost of e' changes, $r(e)$ is unchanged.

According to the property, the affection scope of the link change is bound by the reach of the level. Therefore, the readjustment link set is defined and synchronously changes the reach value, when the cost of link e changes with $UE(e)$, where $UE = \{e' \mid d(e', e) < r^i_{max}\}$ (Figure 4).

The down-up readjustment operation of RHSP is implemented using an iterative algorithm. In a single layer, the reach value of each link around the modified link e satisfying $d(e', e) < r^i_{max}$ is recomputed. If the reach of the link increases and exceeds r^i_{max} (one of the layer's reach bounds) after adjusting, the link is copied to the layer's upper layer. If the reach of the link decreases and drops below r^{i-1}_{max} (one of the layer's reach bounds), the link is assigned to the layer's lower layer.

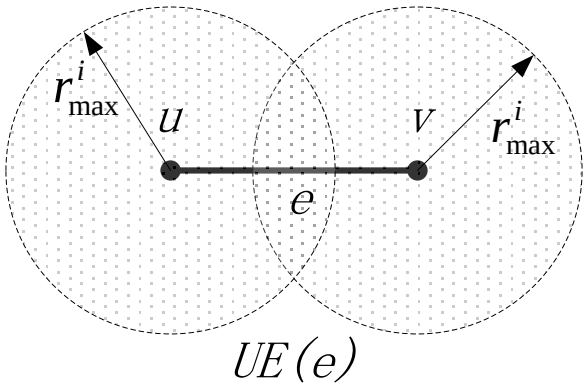


Figure 4. The readjustment link set.

Compared with the hierarchal network artificially defined based on road natural class, the reach-based hierarchical road network (RHN) is consistent due to the readjustment operation after road data updates. Suppose that the time cost of readjustment is $O(\Sigma \mid UE_i \mid)$, and $\mid UE_i \mid$ is the number of link satisfying $d\ (e', e) < r^i_{max}$. The reach bound increases with levels, but the size of the whole network decreases. Therefore, $O(\Sigma \mid UE_i \mid)$ can be described as $O(N \cdot C)$, where N is the total number of RHN, and C is the minimum constant with $C \geqslant \mid UE_i \mid$. That is to say, the readjustment operation is convergent.

2.3. Data Update Mechanism of the CNDF Model

An incremental data update method is used to update local navigation maps when a new data version is received. If a single level of the road network is modified, other levels must be changed to maintain consistent data due to the multi-level design of the network. Therefore, the update methods for navigation data should take into account not only the efficiency of the update mechanism, but also the consistent maintenance after the data update.

2.3.1. The Update Mechanism for Navigation Data

The page is the basic element for the data update. The page is loaded into the memory first, and will be partitioned if its size exceeds the MPS. Then, it is modified and rewritten back to the disk. When being written back to the disk, the modified new page must be inserted into the page chain.

A real-time data update mainly invokes a field update (Figure 5). In most popular page-based models, the length of the field is generally fixed. For a field with fixed length, it only needs to refresh the value of the selected field in the record pointed by the offset table in the page. However, a different strategy is adopted for fields with variable lengths. If the field length exceeds the current field length after the field update, the updated field will be appended to the tail of the right block and the offset table will also be refreshed. In the offset table, "OxFFFF" is a prescriptive tag, representing the field with NULL value.

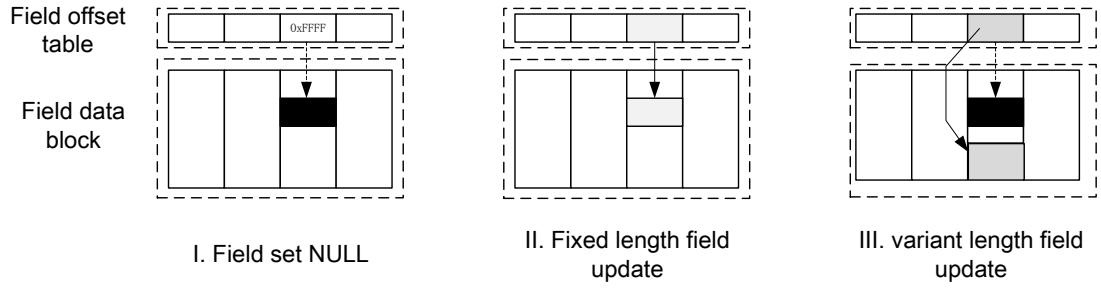


Figure 5. Field update mechanism in real-time data update method.

For models used for spatial data management, it is essential for the data update to first skip the overflow page and then adjust the index tree. In this way, the integrity of the spatial index is maintained, but the computing cost is very expensive. However, it is not suitable for updating navigation maps, due to the small size of the data to be updated and the low update frequency. New version releases of navigation map patches may take several months. Therefore, the CNDF model does not follow this method.

In the CNDF model, if the page overflows after inserting records, the new page containing redundant records will be hooked at the back of the current page (Figure 6). When querying the grid, all pages indexed by the grid (*i.e.*, page 0, page 1, and page 2) are loaded into memory at the same time. It is unnecessary to rebuild the spatial index that cuts off the time cost of the index update, and it is impossible for navigation map patches to concentrate in small number of grids. In this way, the number of pages indexed by each grid will not increase sharply, and the query speed of MSG will not distinctly slow down.

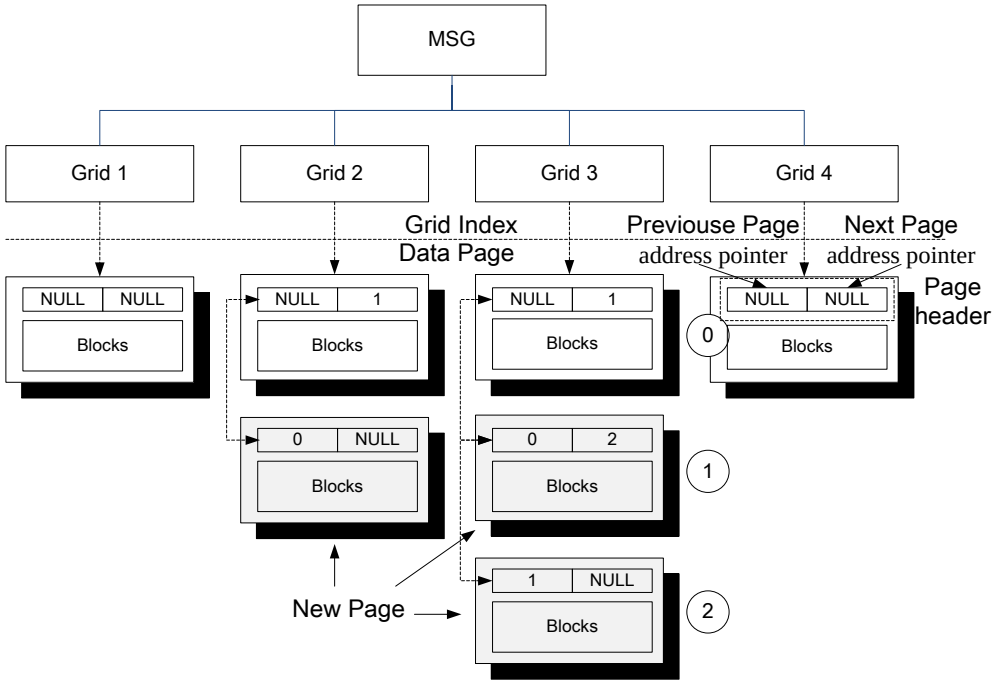


Figure 6. Incremental update mechanism in incremental data update method.

2.3.2. The Maintenance Process for Navigation Data

Data using a multi-level storage pattern are usually redundant. The identity records in each level must be modified simultaneously when updating the original record. The maintenance process means that navigation data, especially the road network, should be consistent after being updated. The first step in the maintenance process is to trace the identity records in each level, and modify them simultaneously when updating the original record. The second step is to change the records related to the updated ones in each level. For example, once the client receives the information that traffic cost of links has changed, the links of *UE* must readjust their reach values to maintain the consistency of the whole RHN. The hieratical mapping is the key technology for tracing the identity record set.

3. Experiment and Results

The navigation map for East China (including Beijing, Tianjin, Shandong, Hebei, and Jiangsu) was used in our experiment. It was built completely using the CNDF model with road classified by the reach metric. Table 3 shows the comparison of the number of links between the natural hierarchy and

the reach hierarchy. To simulate the real vehicle navigation applications, an embedded system was used for our experiment, and the device information is shown in Table 4.

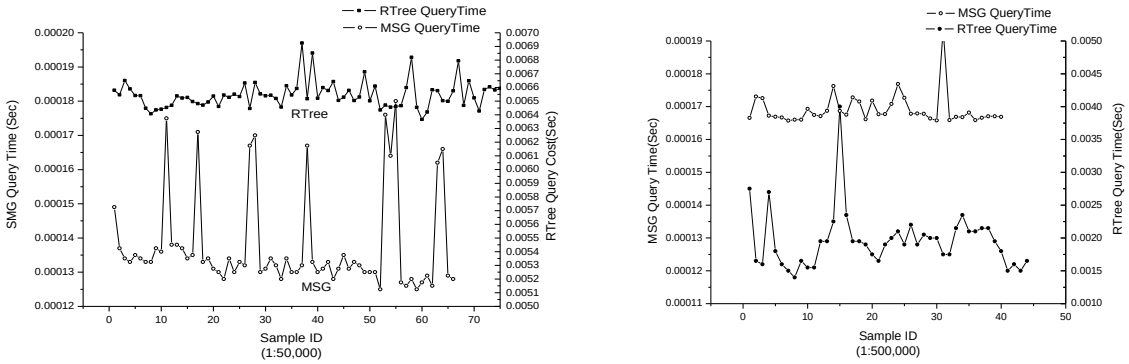
Table 3. The Hierarchy of the Navigation Map for East China.

level	Links of Natural Hierarchy	Links of Reach Hierarchy
0	451,601	329,999
1	53,576	141,908
2	10,701	50,103
3	12,132	6000

Table 4. The Information of the Device used in our Experiment.

Item	Parameter
CPU	ARM 500 MHz
SDRAM	64MB (32Bit)
NOR Flash	64MB (32Bit)
Storage	1GB SD Card (Class 6)
compiler	GCC 3.4.3
Operation system	Linux 2.6

The experiment was to validate the performance of CNDF with respect to operations including spatial query, route search, and hierarchical mapping. To test the efficiency of MGS, the time cost of "QuerybyBound" was continuously recorded by the embedded system, when the maps with 1:50,000 and 1:500,000 were being panned and zoomed, respectively. The operations were then conducted using the map indexed by R-tree. Results show that the average time cost of MGS was less than that of R-tree, and the peak value occurred at the time of switching the levels (Figure 7).



(**a**) (**b**)

Figure 7. The time cost of "QuerybyBound" during (**a**) map pan and (**b**) zoom.

Based on the reach-based hierarchical network, the RHSP algorithm was run iteratively using input start nodes with various distances to the end node. The number of links inserted to the heap was recorded. Results show that the average number of searched links was limited to 11,000 after running 1000 times (Figure 8). This indicated that the reach-based hierarchical network was effective to control the space bound of the route search and constraint the computing time to a nearly constant value.

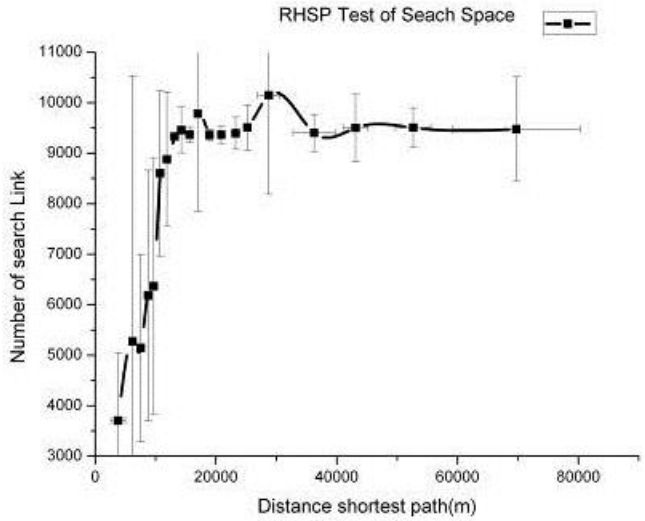


Figure 8. Number of the searched links of the hierarchical shortest-path algorithm (RHSP).

To validate the speed of hierarchical mapping with the LLC algorithm, the real-time traffic information in Beijing was used. The embedded system connected to the information platform of the Beijing Traffic Manage Bureau and received the real-time traffic information through GPRS every 5 min. The peak value of the updated link number was more than 4000 in the cycle of measurement (Figure 9). It was validated that LLC was effective in meeting the requirement of the real-time update for large cities. Considering that the unchanged link did not need to be updated, we restricted the scope of the candidate to update with a time difference strategy, state difference strategy, and composite strategy for real-time traffic information. Figure 9 shows the time cost of the comparative tests under different strategies.

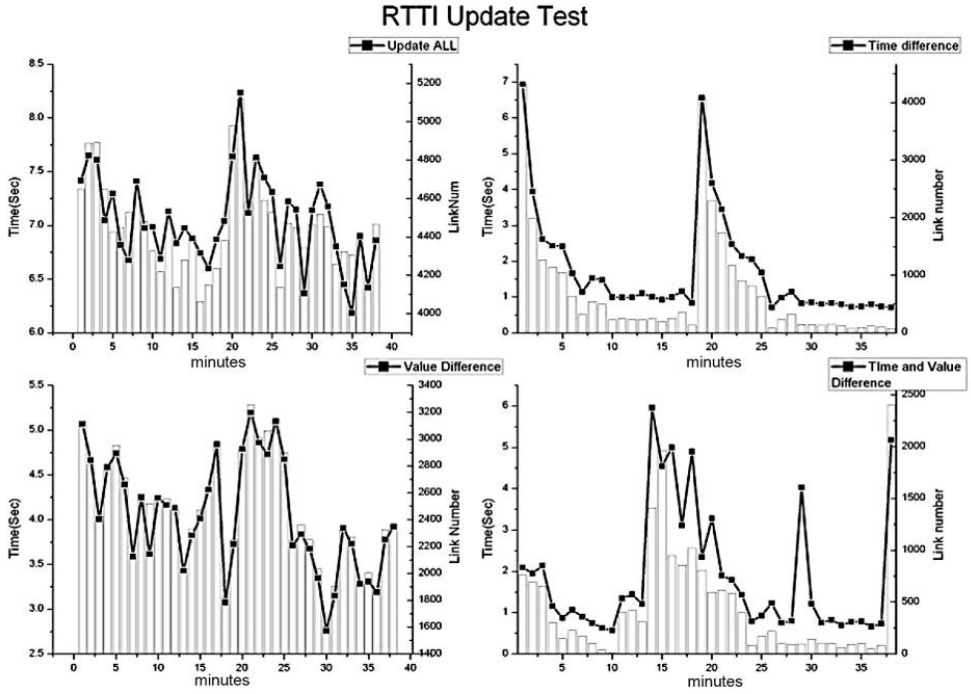


Figure 9. Time cost and the recorded number for updating real-time traffic information (RTTI) every 5 min.

4. Conclusions

A dynamic storage model is very important for vehicle navigation applications. In this paper, we developed a new physical storage format, China Navigation Data Format (CNDF), based on the multi-scale grid index, the reach-based hierarchy approach, and the linear link coding algorithm. Therefore, the CNDF model has a hierarchical and page based structure. It is also a dynamic physical storage format, and the spatial navigation data built by CNDF can be modified in an incremental way. The navigation map of East China built completely using CNDF and the real-time traffic information were used for an embedded navigation system to test the performance of CNDF. Results showed that the time cost, memory cost, and disk storage were dramatically saved. The CNDF would be suitable for the next generation navigation with an intelligent transportation system. However, the hierarchal road network topology update was not considered for the CNDF, which will be studied in further work.

Acknowledgments: Our work was supported by Beijing Science and Technology Special Funds (Z151100003615012), Key projects of Chinese Academy of Sciences (KZZD-EW-07-01-001), the National Key Technology R&D Program (2013BAC03B00), Independent Research Project of the State Key Laboratory of Resources and Environmental Information Systems (088RAC00YA), Surveying and Mapping Public Welfare Project (201512015), Beijing Training Funding for Excellent Talents (201500002685XG242). We also appreciated for the two anonymous reviewers for their constructive comments, which helped improved our work a lot.

Author Contributions: Shaohua Wang and Li Kai conducted the whole work. Ershun Zhong and Guanfu Song provided the idea. Shaohua Wang and Wenwen Cai wrote the manuscript, and response to the reviewers comments.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix

The Linear Link Coding (LLC) Algorithm

The CNDF hieratical mapping model was developed based on the Linear Link Coding (LLC) algorithm. The original record (*i.e.*, the detail level road) should be related to the identity records (*i.e.*, abstract links) to guarantee the consistency of multi-level updates. Considering the compacting storage, the relationship with 1:M was more efficient than that with N:M. Therefore, in this paper, the implicit containing relationship of the link code was used for the LLC algorithm to implement the relationship with 1:M. The definition of the LLC algorithm was as follows.

Definition 1: A linear reference system (LRM) is $\lambda = (\mu, \varepsilon, \xi)$, where λ is the location expression including linear reference method, μ is the linear element, ε expresses distance, and ξ indicates the semantics. In the LLC algorithm, λ is the node post, μ is the link, and ξ is the route distance defined by $\varepsilon_i = \varepsilon_{i-1} + m(i-1, i)/K + 1$ with $\varepsilon_0 = 0$, where $m(i-1, i)$ denotes the distance of link $e(i-1,i)$, K is the predefined length unit, and ε represents the sum of the distance of the coded links.

Definition 2: $e(u, v)$ is a link, and LLC(e) in LRM is a pair of node posts, *i.e.*, <FP, TP>, with $FP = \lambda(u)$, and $TP = \lambda(v)$. LLC(e) is unique, because $\lambda(u)$ is unique in LRM.

Definition 3: The relationship of LLC is given two links e_1 and e_2 with $e_1 \subseteq e_2$, $FP_{e\,1} \geqslant FP_{e\,2}$, and $TP_{e\,1} \leqslant TP_{e\,2}$.

Definition 4: In the consistent hierarchical network, for any link e_i in level i, e_j exists where $T_j \subseteq T_i$ with $j < i$.

According to definition 4, if a hierarchical network is consistent, the relationship between the up level link and the detailed (*i.e.*, low) level link is 1:M. The link code with the up level link code is shown in line 7 in Table T1, which illustrates the LLC algorithm. Therefore, any code of the up level link can contain the code of related low level links, which keeps the consistency of the hierarchical network coded by the LLC algorithm. Using *FP* and *TP*, the abstract links can be queried by the bounds of detailed links with real-time traffic information when updating the data. In this way, the CNDF model implements the 1:M relationship between links dispensed with the array of link identifications.

Table T1. Pseudo Code of the Hierarchal Linear Link Coding (LLC) Algorithm.

FUNCTION Coding
/*RD is the set of detailed road link DRD*/
/*Travel is an abstract link that contains a list of detailed road links*/
FUNCTION Coding (RD)
(1)LM = 0; //Init the code variable
(2)FOR i = MAX_LEVEL TO 1//Code link from top to bottom
(3) FOR EACH DRD IN RD
(4) Travel = {DRD}; //if the level of DRD is less than i or equal to i but has been coded to jump to the next
(5) IF Level (DRD) < i OR Level (DRD) == i AND HasCode (DRD) THEN
CONTINUE;
(6) TraceForward (DRD, i, Travel); //Search next DRD along until meet road crossing
(7) TraceBackward (DRD, i, Travel); //Search prevision DRD conversely until meet road crossing
(8) IF HasCode (DRD) THEN//DRD has been coded in higher level
(9) k = GetLink(UpTravel, Head(Travel), i + 1); //Get the upper level link containing first DRD
Travel.FP = Mid(UpTravel, k).FP; //Assign high link start node post to DRD
k = GetLink(UpTravel, Tail(Travel), i + 1); //Get the upper level link containing last DRD
Travel.TP = Mid(UpTravel, k).TP; //Assign high link end node post to DRD
(10) ELSE//Assign a new coding
(11) Travel.FP = LM
(12) FOR EACH e IN Travel//Assign code to the sub-DRD
(13) e.FP = LM;
(14) LM += Int(m(e)/k + 1);
(15) e.TP = LM;
NEXT s
(16) Travel.TP = LM;
END IF
NEXT DRD
NEXT i

References

1. NAVTECH PSF Specification for SDAL Format Version 1.7. Available online: http://www.janczinsky.cz/dwn/SDAL_spec.pdf (accessed on 5 November 2015).
2. KIWI Format Specification Version 1.2.2 (JIS D0810). Available online: http://www.jsa.or.jp/default_english/default_english.html (accessed on 5 November 2015).
3. Requirements and Logical Data Model for Physical Storage Format (PSF) and Application Program Interface (API) and Logical Data Organization for PSF used in Intelligent Transport Systems (ITS) Database Technology. Available online: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39447 (accessed 5 November 2015).
4. Xu, Y.; Recker, W. Modeling dynamic vehicle navigation in a self-organizing, peer-to-peer, distributed traffic information system. *J. Intell. Transp. Sys. Technol. Plan. Oper.* **2006**, *10*, 185–204.
5. Paz, A.; Veeramisti, N.; Khanal, I.; Baker, J.; Fuente-Mella, H. Development of a comprehensive database system for safety analyst. *Sci. World J.* **2015**, *2015*, 1–14. [CrossRef] [PubMed]

6. Simandl, J.K.; Graettinger, A.J.; Smith, R.K.; Barnett, T.E. GIS based non-signalized intersection data inventory tool to improve traffic safety. In Proceedings of the Transportation Research Board 94th Annual Meeting, Washington, DC, USA, 11–15 January 2015.
7. Alvanaki, F.; Goncalves, R.; Ivanova, M.; Kyzirakos, K. GIS navigation boosted by column stores. *Proc. VLDB Endow.* **2015**, *8*, 1956–1959. [CrossRef]
8. Park, E.; Kim, H.; Ohm, J.Y. Understanding driver adoption of car navigation systems using the extended technology acceptance model. *Behav. Info. Technol.* **2015**, *34*, 741–751. [CrossRef]
9. Feng, J.; Watanabe, T. Index Techniques. In *Index and Query Methods in Road Networks*; Feng, J., Watanabe, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 11–39.
10. Wu, H.; Liu, Z.; Zhang, S. Spatial-Temporal dynamic segmentation model. In Proceedings of the 10th International Conference of Chinese Transportation Professionals, Beijing, China, 4–8 August 2010.
11. Wagner, D.; Willhalm, T. Speed-Up techniques for shortest-path computations. In *STACS 2007: 24th Annual Symposium on Theoretical Aspects of Computer Science*; Thomas, P., Well, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 23–36.
12. Chen, C.; Jia, Y.; Shu, M.; Wang, Y. Hierarchical adaptive path-tracking control for autonomous vehicles. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 1–13.
13. Li, Q.; Zeng, Z.; Zhang, T.; Li, J.; Wu, Z. Path-finding through flexible hierarchical road networks: An experiential approach using taxi trajectory data. *Int. J. Appl. Earth Obs. Geoinform.* **2011**, *13*, 110–119. [CrossRef]
14. Weng, M.; Jiang, S.; Qu, R. Hierarchical spatial reasoning and case of way-finding. *Geo-spatial Inf. Sci.* **2008**, *11*, 269–272. [CrossRef]
15. Schubert, E.; Zimek, A.; Kriegel, H.P. Geodetic distance queries on r-trees for indexing geographic data. *Lect. Notes Comput. Sci.* **2013**, *8098*, 146–164.
16. Delling, D.; Sanders, P.; Schultes, D.; Wagner, D. Engineering route planning algorithms. In *Algorithmics of Large and Complex Networks: Design, Analysis, and Simulation*; Lerner, J., Wagner, D., Zweig, K.A., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 117–139.