IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 13, 2019, accepted March 19, 2019, date of publication March 22, 2019, date of current version April 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2906871

A Geographic Meshing and Coding Method Based on Adaptive Hilbert-Geohash

NING GUO, WEI XIONG, YE WU, LUO CHEN, AND NING JING
College of Electronic Science, National University of Defense Technology, Changsha 410073, China

Corresponding author: Luo Chen (luochen@nudt.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 41871284.

ABSTRACT Geographic meshing system is an essential technology in the digital earth framework and has essential applications in the integration and organization of heterogeneous spatial data, along with corresponding coding method. But current meshing and coding methods show unsatisfying locality and performance. To make an improvement, an adaptive Hilbert–Geohash meshing and coding method called AHG is proposed, which could represent both the location and the approximate size of the coded object directly by the meshing hierarchy and the corresponding coding length. This unique feature helps to accelerate the spatial range query and neighbor query. By simple string operations, many candidate objects that do not meet the query criteria can be quickly filtered out without precise spatial calculation. In addition, AHG code can also support spatial size query by finding objects whose size is within a certain interval quickly, without calculating the precise size of each object, which brings great convenience to spatial size statistics of a massive spatial dataset. Demonstrated by experiments over different types of the spatial dataset in a common PC environment, AHG shows favorable stability and scalability besides its capability in accelerating spatial query. The method is now applied successfully in several spatial query tools in a high-performance geographic information system called HiGIS.

INDEX TERMS Adaptive encoding, geohash, meshing, pre-filtering, spatial query.

I. INTRODUCTION

The geographic meshing system, also known as the geospatial reference grid system, divides the surface of the earth into a frame of multi-level grids without spatial overlapping. Usually, each grid is coded successively by a recursion order of the geographic mesh [1], [2]. The goal of the meshing system is to integrate geospatial positioning and geographic feature descriptions into the grids ranging from centimeter scale to the global scale [2]. Along with the meshing model, a geographic grid system also involves a grid coding method. An appropriate coding method can serve as simple grid index which helps to improve spatial retrieval efficiency through dimensional reduction. Moreover, grid codes can also serve as a reference for spatial partitioning, which is essential to distributed data management for massive dataset. Reference [35]–[37] are of such applications in which geohash-based index are proposed to accelerate the retrieving and processing for big data both in distributed memory and in Hbase.

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

In order to form the earth sphere, many global geodesic meshing systems use a combination of regular polygons to stimulate the surface of the earth [3] just like the soccer surface. Some well-known global meshing systems include the hexagonal based DGGS [3] and degenerated octree 3D-grid [4]. However, these methods of subdivision using regular polygons bring additional complexity to the coding transformation, grid computing and position description for spatial objects.

In terms of meshing and coding method, researchers often use the geographic coordinates of the grid to implement the process in a hierarchical way. Typical methods include Geohash proposed by Morton [5] and GeoSOT proposed by Song *et al.* [6], Sun and Cheng [7]. There are many research about the application of geohash-based index and the rapid retrieval of big spatial data using the classical meshing and coding method like [28]–[32], [34]–[38]. There are also some research focus on improving the efficiency of geohash encoding, such as [39] implemented a method of encodeing lat/lon coordinate to Geohash code on FPGA architecture. However, most of these methods only support the encoding of the coordinates of a point object at a specified mesh level, which is not compatible with line or surface object because the input of traditional geohash encoding is a single pair of latitude and longitude. For 2D object, certainly we could simply encode the up-left and the down-right point of the object and get the longest matching prefix as the coding result. But the strategy has a non-negligible shortcoming. For example, if a line or polygon object spans a primary level grid boundary, its matching prefix's length will be extremely short, which will introduce great spatial error when describing the location of the object. Moreover, traditional Geohash adopts Z-order curve as the coding order. Z-order curve shows frequent coding mutations at certain locations, which we will explain in detail by an example in Methodology section.

To address the problem, we propose an adaptive Hilbert-Geohash (AHG) meshing and coding method which can adaptively adjust the hierarchical grid level according to the size of the spatial object. Moreover, it uses the Hilbert curve to encode grid to get better spatial locality, which means the grid code of adjacent grids is basically adjacent. AHG is easy to implement and can provide more accurate spatial pre-filtering for spatial range query and spatial neighbor query. The coding method can also support a new type of spatial query called spatial size query by screening out objects whose spatial sizes is within a certain interval efficiently without calculating the size of them. All these features are of practical application value.

II. RELATED WORK

A. MESHING MODEL

Discrete global grid and meshing systems is an important aspect of digital earth framework. Its basic elements include cell shape, initial discretization, refinement, projection, cell indexing and rendering [1]. The global geographic grid subvision scheme is mainly classified into two categories. One is the combination of regular polygon mesh trying to fit the surface of the earth, represented by DGGS; The other is based on the geographical plane projection coordinates which mainly consists of multi-level rectangular grid.

In recent years, a variety of polygonal meshing schemes have appeared in the academia, such as HEALPix [8], Ellipsocial Cube Map and SCENZ-Grid [9], Crusta [10], Dutton's Quaternary Triangular Mesh (QTM) [11] and Goodchild's Hierarchical Spatial Data Structure (HSDS) [12], which are mainly based on cell fitting of triangle or quadrilateral. There are also methods based on hexagonal cell fitting like Icosahedral Snyder Equal Area Aperture 3 Hexagon (ISEA3H) DGGS and PYXIS Indexing [13], [14], Sahr's Central Place Indexing (CPI) [15] and Hexagonal Quaternary Balanced Structure (HQBS) [16]. Diversified polygon combinations and indexing methods have achieved more fitting accuracy of Earth's surface [3], [17], [18]. But these solutions are hard to implement and have high application barriers. As a result, they mainly stopped at the level of theory research and model description.

Most of the mature products like Google Map, Google Earth, Bing map and Cesium use rectangular meshing scheme, such as simple quadtree partitioning, CDB, C-squares [19], Clip-Maps [20]. They implement efficient meshing and coding at the cost of small precision lost and make effective use of their grid system in spatial data organization, spatial query and rapid visualization.

Recently, three-dimensional spherical subvision models have also been researched [18], such as Spheroid Degenerated-Octree Grid (SDOG) [21], which can directly realize the stereoscopic division of Earth space. However, the high computational complexity restricts its application.

B. GEOGRAPHIC GRID STANDARD

The geographic grid plays a fundamental role in the 9 of spatial data and various professional organizations in major countries have developed their own geographic grid standards. USNG (United States National Grid) [22], the world Geographic reference system GeoRef [23], China's GNGC (Global Navigation Grid Code) and World Geographic Grid standard [24] are some of the well-known standards. These grid systems have similar structure in which they expand the surface of the earth into a plane according to a specified projection. Then the plane is spit both horizontally and vertically to form different levels of rectangular space grid of a different density, and each level of grid is coded according to a certain traversal order. There are also a variety of coding systems designed with reference to these standards in specific applications, such as Military Grid Reference System (MGRS) [25] and the Global Area Reference System (GARS) [26]. MGRS divides the original earth surface into $8^\circ \times 6^\circ$ grids, and then continue to split them into different levels of rectangular grids according to Universal Transverse Mercator (UTM) projection. The rectangular grid size ranges from 100km, 10km, 1km and even finer. GARS divides the global surface into three levels of grids in 30', 15' and 5'. Apart from them, the Open Location Code [27], Geohash [5] and GeoSOT [6], [7] organize the grid like quadtree tile, which divides the surface into two parts recursively and get the incremental grid codes synchronous.

In summary, the general idea of geographic grid coding is consistent: any position on the earth's surface can be represented by a specific grid and corresponding code. The differences lie in the grid division patterns and coding rules. Among the diverse solutions, Geohash coding has simple rules, controllable position describing precision and good data compatibility. Furthermore, it can be theoretically proved that Geohash mesh is isomorphic with other rectangular-based global geodesic meshes and existing grid standards. It has been extensively used in geographic information organization and spatial data index [28]. There are plenty of research and achievements in the use of Geohash coding for KNN query and trajectory data organization [29], [30]. The weakness of Geohash is that there is no clear rule for the encoding of multi-dimensional objects like line or polygon [20], [31]. Moreover, since the coding order of the original Geohash grid is equivalent to Z-order curve, as shown in Fig. 1, its spatial locality is not good. Namely in some cases, the geohash codes of nearby points may have great difference [32], which brings obstacles in spatial application.

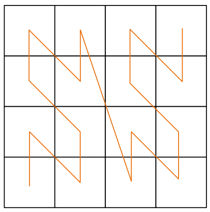


FIGURE 1. Z-order space filling curve.

III. METHODOLOGY

In this paper, an adaptive Hilbert-Geohash (AHG) meshing and coding method is proposed. It can adaptively adjust the hierarchical level of the mesh according to the size of the spatial object and uses the Hilbert space filling curve to traverse the grids at that level. Hilbert space-filling curve brings better spatial locality [32] and the automatic adjustment algorithm can make the Geohash code length inversely proportional to the size of spatial object. In other words, the smaller the object size is, the longer the code length and the higher the coding precision will be. Both the improved features provide better support for spatial objects organization and help to broaden the application range of simple meshing system or simple spatial grid coding method.

A. PRELIMINARY KNOWLEDGE

Geohash is a geocoding technology based on geodetic coordinates. It uses a character string to represent the location of a spatial point object [5]. It continuously bisects the global space and uses 0 and 1 as the two-part space code. Gradually it reduces the spatial scope to the target point and converts the precise two-dimensional coordinates into a one-dimensional string. The grid division example of the first two hierarchies is shown as Fig. 2:

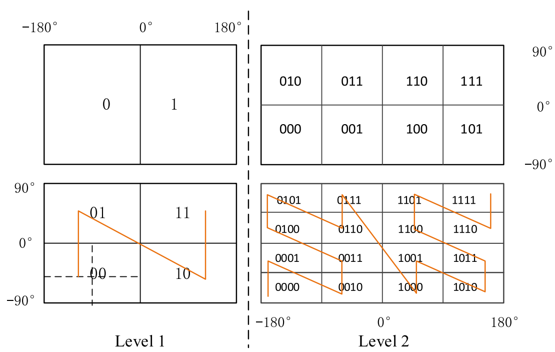


FIGURE 2. Geohash binary code.

According to the coding principle of Geohash, it divides the surface of the earth into multiple levels of grid through continuous bisection and each level of grid consists of arrays of rectangular in a specific size. As for point objects, Geohash coding can be performed directly and the coding length depends on the accuracy requirement. As for linear or polygon object which can be represented by the minimum bounding box (MBR), a common solution is to perform Geohash coding on the upper left corner and the lower right corner of the object's MBR, and then use the longest matching prefix of the two codes to be the Geohash code of the object. However, when the object's MBR cut across a low level Geohash mesh's border, the length of match prefix may be too short to locate the object effectively. For example, a line segment consisting of (−0.0001°, −0.0001°) and (0.0001° , 0.0001°) has a distance of only tens of meters, but the two endpoints' Base32 [33] Geohash codes are 7zzzzzzzmtm7 and s0000000d6ds which is a phenomenon of mutation. There is not any common prefix between them. Therefore, in order to apply Geohash encoding to linear and polygon object, it is necessary to make special improvements to the Geohash coding method to solve the problem.

Compared to traditional Geohash encoding method, our Adaptive Hilbert-Geohash (AHG) coding method adopts Hilbert space-filling curve shown in Fig. 3 to sort the mesh grids, which shows better spatial locality for neighbor objects.

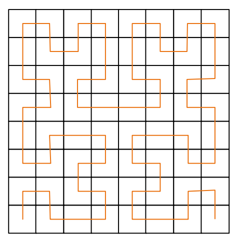


FIGURE 3. Hilbert space filling curve.

B. CODING PROCESS

Firstly, depended by the size of the spatial object, the coding level is calculated by a circular judgement. In this level, the size of a single grid is exactly no less than the spatial extent of the spatial object's MBR. Then the Hilbert-Geohash [32] at this level is conducted based on the centroid of the MBR. The final result will be the AHG code of the current spatial object. The overall flowchart of the coding process of AHG encoding is shown as Fig. 4.

We use a recursive method to get the adaptive coding level to compute the Geohash code based on Hilbert space filling curve, which expand the application of [32]. The AHG coding result can reflect the spatial position of the spatial object to an adaptive precision, while the code length reflects the approximate size of the object. The method has better spatial locality than the traditional Geohash [32] and can support encoding of line and polygon objects without the problem of a large object corresponding to a short geohash code.

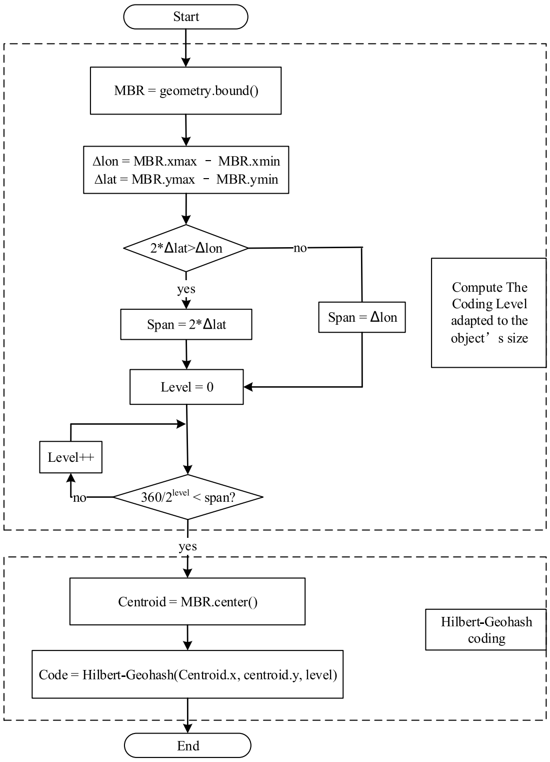


FIGURE 4. Flowchart of AHG coding.

C. CODING EXAMPLE

In this part, we take a typical polygon object Lake Superior as an example to show the AHG method.

Lake Superior is the biggest freshwater lake in the world whose MBR is as follows:

$$lon_{min} = -92.25080118^\circ$$
$$lat_{min} = 46.41255453^\circ$$
$$lon_{max} = -84.33279621^\circ$$
$$lat_{max} = 48.99994719^\circ$$

Therefore:

$$\Delta lon = lon_{max} - lon_{min} = 7.91800379^\circ$$
$$\Delta lat = lat_{max} - lat_{min} = 2.58739266^\circ$$

And the coordinate of centroid point is (−88.29179870°, 47.70625086°)

Using the formula below to calculate the adaptive coding level:

$$\frac{360}{2^{l_{adapt}+1}} < \max\{\Delta lon, 2 \times \Delta lat\} \leqslant \frac{360}{2^{l_{adapt}}}, \; level \in \mathbb{N}^{+}$$

Solving the inequation, we can get $l_{adapt} = 5$. So the binary code's length is 10 bits. Then we use the centroid point of the MBR as the encoding input and use Hilbert space-filling curve as the order of traversing the 5-level grids. Finally, we could get the AHG code of Lake Superior: *0110000000*. The location error of this geo-code level is around 625 kilometers, which is approximate to the size of this grand lake. The result indicates that AHG code can reflect both the location and size character of 2D spatial objects. The process of gradual refinement of the mesh subdivision is shown by Fig. 5.

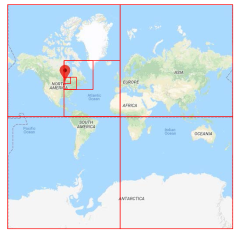


FIGURE 5. Mesh subdivision of 5^{th} level when encoding Lake superior.

IV. TYPICAL APPLICATION

The classical organization method for spatial data which contain complicated geometry is to load the datasets to a spatial database as a two-dimensional table. After that a spatial index such as R tree can be constructed on the spatial attribute column to support the user's spatial query. With the development of distributed architecture and cloud computing, distributed file system is now commonly used to store massive dataset. It provides a way of managing and organizing large scale data efficiently. By AHG meshing and encoding method, spatial data stored in file system can be directly organized by the adaptive code without using spatial database. In fact, the AHG can serve as a simple spatial index and can be used as the reference of data partitioning. More importantly, AHG code can support direct spatial query including spatial range query and spatial neighbor query. By only a few simple string operations, most of the candidate objects can be filtered out before performing accurate spatial calculation on them Moreover, AHG code can be used to support a new type of query called spatial size query. Without calculating the precise size, the object who's the size is within a certain interval can be screened out quickly according to the length of the AHG code. The following sections will emphasize on these typical applications of AHG.

A. SPATIAL RANGE QUERY

Spatial range query generally refers to finding the spatial object that intersects with the query box. The query box's AHG code can be used to filter out objects efficiently that do not meet the query criteria. Since the centroid of the spatial object is distributed in space and the Geohash mesh boundary in a given level is fixed, there may be cases where the spatial object intersects the query box but the centroids of the two are not in the same grid [34]. Just As showed in Fig. 6, a linear object L in the dataset intersects the query box, but the centroid of L's MBR is located below the grid where the query box is located. So, it is necessary to expand one grid around the query grid and then use the AHG codes of them to perform the filtering operation. The relationship between the query box and the grid can be divided into the following three cases:

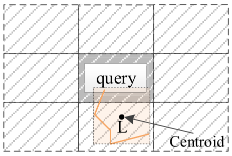


FIGURE 6. Query box's MBR is contained in one grid.

1) The MBR of the query box falls completely within one grid on the adaptive level as it shows in Fig. 6. When doing spatial query through AHG code, we need to expand the eight adjacent grids as query grid, which is shown in the shaded part of the figure.

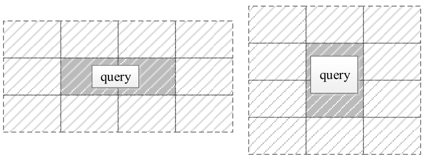


FIGURE 7. Query box's MBR intersects with two grids.

2) The MBR of the query box falls into two grids on the adaptive level. It includes two scenarios presented in Fig. 7. The centroid is located in one of the two grids. But the AHG code of the candidate object of query can be corresponded to one of the two grids or the adjacent grids. Therefore, the query scope has to be extended to the shaded part like the figure shows.

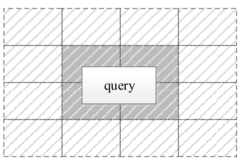


FIGURE 8. Query box's MBR intersects with four grids.

3) The MBR of the query box falls within four grids on the adaptive level. In this case, the query scope is extended to the shaded part shown in Fig. 8.

The general query steps are as follows:

1. Perform the AHG encoding operation on the query box to obtain the query grid and the adaptive coding level of the box (denoted as m), and calculate the size of the grid of the m level.

2. Extend the query grid to the adjacent ones. Then perform the Hilbert-Geohash encoding for them on the level m, getting a set of Geohash code strings in length of m as the extended query code set.

3. Traverse the AHG code of all spatial objects in the spatial dataset to perform string-matching judgement with the extended query code. The matching rule can be divided into two cases:

a) If the AHG code length of the spatial object is no less than m, the matching rule is that the first m bits of the code are in the extended query code set;

b) If the AHG code length of the spatial object is less than m, it is necessary to perform m-level Geohash coding on the centroid of the object to obtain a m-bit string, and the rule is that the m-bit string is in the query code set.

In this way, most of the objects that are not intersected with the query box can be filtered out, and a rough result of the range query is obtained.

4. The final result of the range query can be obtained by traversing the rough result set to perform accurate spatial topology judgement.

The pseudo code of the filtering is as follows:

Algorithm 1 Spatial Range Filtering Using AHG Encoding

Input: spatial data set (**S**), spatial query box (**Q(X$_{\mathbf{left}}$, X$_{\mathbf{right}}$, Y$_{\mathbf{down}}$, Y$_{\mathbf{up}}$)**).
Output: Streamlined set for precise spatial relation judgement.
C_q=AHG(Q) // AHG code of the query box
m =length(C_q) // The adaptive mesh level m of the query box;
Extend=[], Result = [] // Initialize the extended query code set and result
P_{lu}= Point(X_{left}, Y_{up}), C_{lu}= Hilbert-Geohash(P_{lu}, m)
P_{rd}= Point(X_{right}, Y_{down}), C_{rd}= Hilbert-Geohash(P_{rd}, m)
P_{ld}= Point(X_{left}, Y_{down}), C_{ld}= Hilbert-Geohash(P_{ld}, m)
P_{ru}= Point(X_{right}, Y_{up}), C_{ru}= Hilbert-Geohash(P_{ru}, m)
If *C_{lu}== C_{rd}: // The query box MBR is in a grid*
  Extend.append(C_q)
Else:
  If *C_{lu}== C_{ld} or C_{lu}== C_{ru}: // Query box spans 2 grids*
    Extend.append([C_{lu}, C_{rd}])
  Else: *// Query box spans 4 grids*
    Extend.append([C_{lu}, C_{rd}, C_{ld}, C_{ru}])
For *N in neighbor(extend): // neighbor points of grids in extend*
  C_n= Hilbert-Geohash(N)
  Extend.append(C_n)
For *obj* **in** *S* :
  L= length(C_{obj}) // length of obj AHG code
  If *L >= m:*
    If *C_{obj}[0:m] in Extend:*
      Result.append(obj)
  Else:
    $C_{obj-m} = Hilbert - Geohash(obj, m)$
    If *C_{obj-m}in Extend:*
      Result.append(obj)
Return *Result*

B. SPATIAL NEIGHBOR QUERY

Spatial neighbor query usually refers to finding the object whose distance to the target point is within the given threshold and there are many proposed methods or algorithms of accelerating the query process. Special designed index structures and pruning strategies are common solutions[40], [41]. Geohash has already been widely used in neighbor queries, especially kNN query [32], but due to the drawback in representing the line and polygon objects, Geohash can only support neighbor query for point dataset directly. AHG coding method makes up for this shortcoming. The steps of neighbor query using AHG code are as follows:

1. Calculate the hierarchical level n of the Geohash mesh according to the given distance threshold, and compute the size of this mesh, denoted as a, b.

2. In order to keep all eligible objects in result, we need to extend the query scope to the eight neighborhoods of the target point.

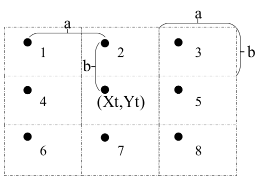


FIGURE 9. Query box's MBR intersects with four grids.

As shown in Fig. 9, according to the spatial relationship, the eight grids around the target point contain the following 8 neighbor points:

$$(x_t - a, y_t - b), (x_t, y_t - b), (x_t + a, y_t - b),$$
$$(x_t - a, y_t), (x_t + a, y_t),$$
$$(x_t - a, y_t + b), (x_t, y_t + b), (x_t + a, y_t + b)$$

3. Perform Hilbert-Geohash coding for original target point and the 8 neighbor points on the level n and get the code set as the query reference.

4. Traverse the AHG code of all objects in the spatial dataset, and perform string-matching judgement with the query code set. The matching rule can be divided into two cases:

a) If the AHG code length of the spatial object is no less than n: the matching rule is that the first n bits of the code are in the query code set.
b) If the AHG code length of the spatial object is less than n: perform n-level Geohash coding on the centroid of the object to obtain an n-bit string, and the rule is that the n-bit string is in the query code set.

In this way, most of the objects that do not meet the neighboring conditions can be filtered out, and a rough result of the neighbor query is obtained.

5. Calculate accurate distance between the target point and the object in the rough result to get the final result.

The pseudo code of the first 4 steps is as follows:

Algorithm 2 Spatial Neighbor Filtering Using AHG Encoding

Input: spatial data set (**S**), target point ($\mathbf{T(X_t, Y_t)}$), distance threshold (**d**)
Output: Streamlined set for precise distance calculation.
n = 1
While $180/2^{i-1} > d$:
  n =n + 1 // coding level corresponding to the threshold
$a = 180/2^{n-1}$, $b = 180/2^{n-1}$ // the size of grid in level *n*
C_t= *Hilbert-Geohash(T, n)*
C_1= *Hilbert-Geohash(* $(x_t - a, y_t - b)$ *, n)*
C_2= *Hilbert-Geohash(* $(x_t + a, y_t - b)$ *, n)*
C_3= *Hilbert-Geohash(* $(x_t + a, y_t - b)$ *, n)*
C_4= *Hilbert-Geohash(* $(x_t - a, y_t)$ *, n)*
C_5= *Hilbert-Geohash(* $(x_t + a, y_t)$ *, n)*
C_6= *Hilbert-Geohash(* $(x_t - a, y_t + b)$ *, n)*
C_7= *Hilbert-Geohash(* $(x_t, y_t + b)$ *, n)*
C_8= *Hilbert-Geohash(* $(x_t + a, y_t + b)$ *, n)*
querySet = *[*C_t, C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8,*]*
Result = *[]*
For *obj* **in** *S* :
  L= length(C_{obj}*) //length of obj's AHG code*
  If *L >= m:*
    If C_{obj}*[0:m] in Extend:*
      Result.append(obj)
  Else:
    C_{obj-m} *=Hilbert-Geohash(obj, m)*
    If C_{obj-m}*in Extend:*
      Result.append(obj)
Return *Result*

C. SPATIAL SIZE QUERY

AHG coding can also support a new type of query called spatial size query. We take the longer side length of the MBR of the spatial object to roughly represent the overall size of the spatial object. Utilizing the AHG code's ability of expressing object size, we can effectively screen out the object whose size is within a given interval without calculating the precise size of every spatial objects.

The implementation is quite simple. Just convert the given size interval into a hierarchical interval of the Geohash mesh and then traverse the candidate dataset to filter out the objects whose AHG code length is within the interval. This method greatly reduces the computational complexity. The pseudo code of the algorithm is as follows:

V. EXPERIMENTS

In order to verify the efficiency of spatial query based on AHG code, we implement the AHG coding method and spatial filtering algorithms in Python and carry out the experiments on OSM datasets of different scales and different spatial types. The datasets contain various thematic types, such as poi, road, water, building, and the object quantity varies from 10^2 to 10^6.

Algorithm 3 Spatial Size Filtering Using AHG Encoding

Input: spatial data set (**S**), query size interval (**[d1, d2]**)
Output: Streamlined candidates for precise size calculation
m = 1, n = 1 // initialize the maximum and minimum of grid level
While $180/2^{i-1} > d1$:
  m = m + 1 //compute maximum of mesh hierarchy
While $360/2^{i-1} > d2$
  n = n + 1 ////compute minimum of mesh hierarchy
Result = []
for *obj* **in** *S* :
  $l = length(C_{obj})$
  if *(n-1) < l <= m:*
    Result.append(obj)
return *Result*

TABLE 1. Experimental dataset.

Dataset	point	line	polygon
Beijing osm	9760	100465	56013
China osm	336663	1982465	1183975

TABLE 2. Experimental environment.

item	Description
Operating System	Ubuntu 16.04 LTS
CPU	Intel® Core™ i7-4710MQ CPU @ 2.50GH
RAM	16 GB
File System	HDFS 3.0
Program language	Python 3.6

A. CODING EFFICIENCY

For different types of spatial datasets, AHG coding is tested on different scales of input data. As a comparison, we implement the traditional geohash encoding on the same environment. The data source description and encoding time costs of the two methods are shown in the following tables:

TABLE 3. Encoding experiment of point datasets.

	Point Data	Obj-Quantity	AHG/s	Traditional/s
1	amenity_bj	472	0.217	0.198
2	place_bj	635	0.132	0.114
3	barrier_bj	1603	0.224	0.122
4	pofw_cn	2192	0.531	0.36
5	transport_bj	7050	0.625	0.337
6	transport_cn	54913	5.354	3.492
7	pois_cn	86346	7.024	3.673
8	place_cn	127346	11.417	5.825

To demonstrate the results more vividly, we extract the object quantity and the encoding times to line-bar charts below.

It can be seen that AHG costs a little more time when encoding spatial object than traditional Geohash. This is due to the additional adaptive encoding progress and the improved method of spatial order of traversal. It is the necessary price

TABLE 4. Encoding experiment of line datasets.

	Line Data	Obj-Quantity	AHG/s	Traditional/s
1	aeroway_bj	608	0.284	0.264
2	waterway_bj	679	0.296	0.162
3	barrierway_bj	1206	0.227	0.222
4	road_short_bj	31144	3.231	1.67
5	road_long_bj	66828	9.011	5.799
6	waterway_cn	95049	26.108	23.926
7	railway_cn	122399	15.153	12.498

TABLE 5. Encoding experiment of polygon datasets.

	Polygon Data	Obj-Quantity	AHG/s	Traditional/s
1	transport_bj	129	0.165	0.152
2	natural_cn	4819	2.93	1.089
3	traffic_cn	16084	3.309	1.987
4	Building_bj	47840	6.228	3.673
5	pois_area_cn	62227	9.209	6.723
6	water_area_cn	107255	32.411	29.85
7	Landusage_cn	237839	46.532	41.951
8	Building_cn	747354	97.302	68.896

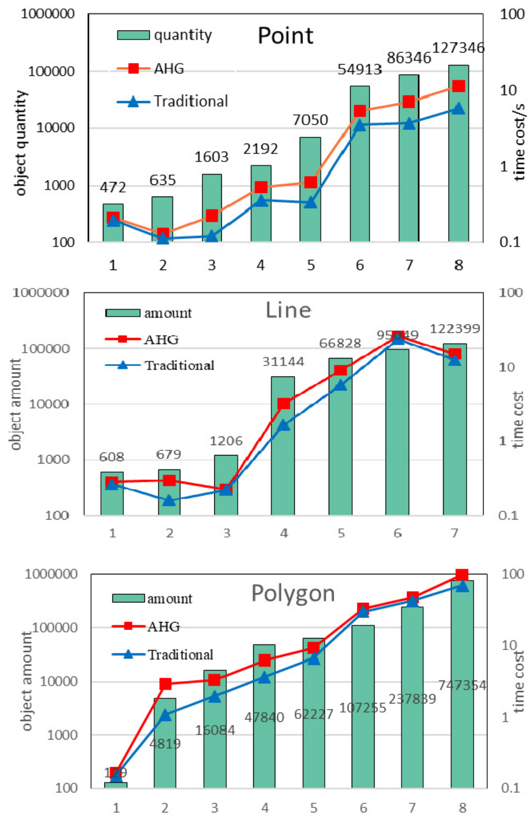


FIGURE 10. Code efficiency of AHG and traditional geohash.

for more practical applications and better query performance. For the three types of spatial data, as the number of objects increases from 100 to 100,000, the time cost of both AHG and traditional Geohash encoding increases linearly. The only conspicuous exception is the encoding of line dataset 6 and 7. They are waterway and railway objects. The number of railway dataset has much more object than the waterway's, but its encoding time is much shorter. The abnormal result is due to the different spatial characteristics in that the waterway objects represent rivers or other water bodies which contain many twists and turns, so each waterway line contains more line nodes. It results in an increase in the amount of calculation in the encoding process. The railway object is much simpler and contains fewer nodes. It indicates that the timecost of AHG encoding, similar to traditional Geohash encoding, is positively correlated with both the number of objects in the dataset and the number of spatial coordinates contained in the object.

B. SPATIAL QUERY FILTERING

We select three typical spatial datasets of different types to be the validation data:

TABLE 6. Experimental dataset sample.

Type	point	line	polygon
Dataset	china_places	china_railways	china_buildings
Obj-Quantity	127346	122399	747354

1) RANGE QUERY

Spatial range query is the most typical type in spatial inquiry and we can use AHG code's ability of representing spatial location to accelerate the process. To demonstrate it, we use query boxes of different sizes to query spatial datasets of different types based on AHG code. The query box size ranges from 0.001° to 50°, which could approximately represent the size from a street block to a continent. We recorded the hit ratios and average time costs. Because traditional Geohash method can only encode point object, we implement range query of point dataset based on traditional Geohash code in the same environment as a comparison. The results are shown by the following line-bar charts:

It can be seen that in the three kinds of query, AHG code shows good pre-filtering effect. As the size of the spatial query box increases, the hit ratio also rises gradually, and the query time cost grows smoothly. It indicates that AHG code can screen a rough result at an acceptable time expense.

In the comparison experiment of point query, we could see than the traditional Geohash result has much more candidate left after code-based pre-filtering than AHG, which means AHG can filter out much more candidates than traditional Geohash code at an approximate timecost. For large scale query, AHG performs even better.

In addition, the data source of the line query is the railway linestring of China and the length of the railway data is usually longer than 0.001 rad (about 110 meter) due to the mapping approach. So, when query box size exceeds 0.001 rad, the candidate is filtered out by the code length, and more processing time is needed to do string matching operation. Therefore, there is a slope change at the beginning in the second figure.

2) NEIGHBOR QUERY

To demonstrate AHG code's ability of pre-filtering in spatial neighbor query, we use different distance thresholds to do neighbor query on spatial data of different types. The values of distance threshold range from 10m to 5,000,000m. The hit ratio and time cost are shown as Fig. 12. And similarly, because traditional Geohash method can only encode point object, the compare experiment is only carried out on point dataset.

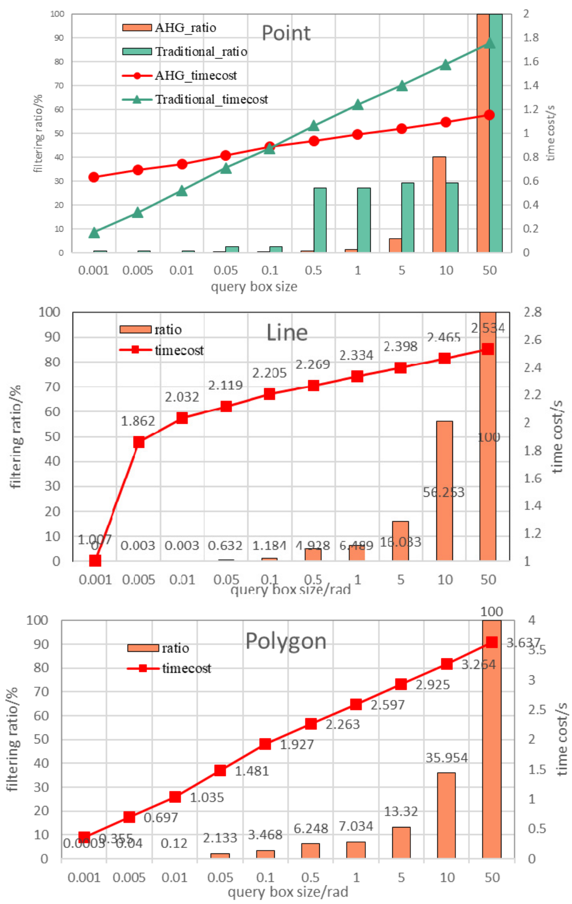


FIGURE 11. Range query's hit ratio and time cost.

It can be seen that analogous to the spatial range query, as the distance threshold increases, the hits ratio of the query rises, and the query takes longer on the average. For the same query distance, the time-consuming order is: *point object* < *line object* < *polygon object*, which is consistent with the conclusion that the AHG code is proportional to the number of coordinates contained in the spatial object.

In the comparison experiment of point neighbor query, we could draw similar conclusion with the range query experiment. AHG can filter out much more candidates than traditional Geohash code. And as the distance threshold is larger than 10,000 meters, AHG has better performance than traditional Geohash code.

As mentioned above, the experiment of line object is based on the China railway data. When the query threshold reaches 100 meters, more candidate objects are processed in the string-matching step. So, the time cost shows a sudden rise at the beginning in the second figure.

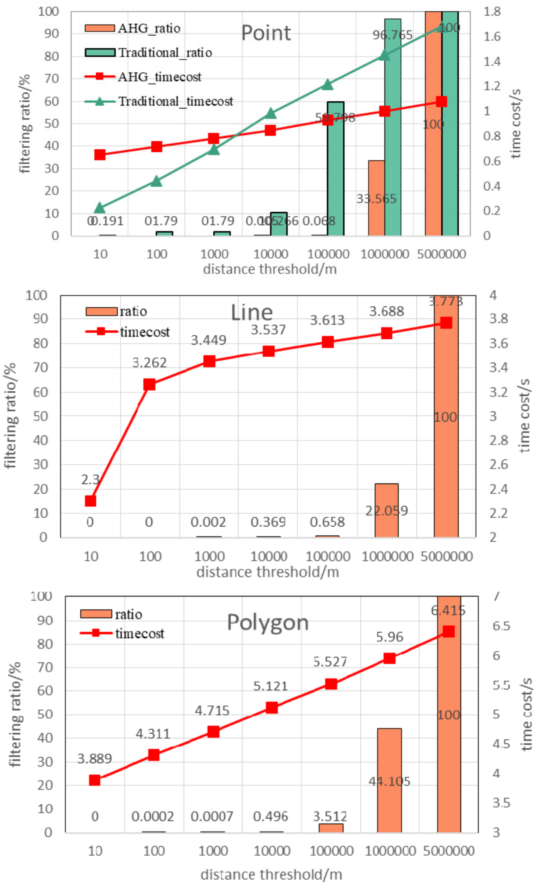


FIGURE 12. Neighbor query's hit ratio and time cost.

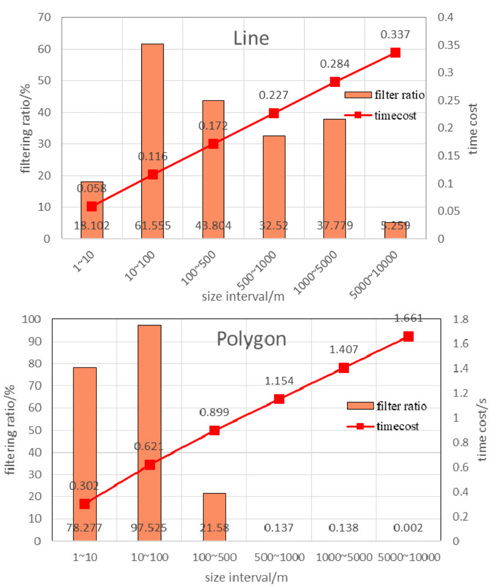


FIGURE 13. Size query's hit ratio and time cost.

3) SIZE QUERY

Size query is a new type of query that AHG can support. We can get the size distribution of 2D spatial datasets in a rough way using AHG code. The experiment is carried out on China railway and building dataset, which contains 122,399 linear objects and 747,354 surface objects. We select different intervals of size and get different hit ratio and the result are shown below.

By the spatial size filtering, we can excavate some useful information quickly. For example, the first chart reveals that, in OSM dataset of China, most of railway line's length is within 10m to 5000m, which probably results from both the spatial characteristics of China's railway and the data collecting habit of OSM contributors. From the second chart, we could find that the dimension of buildings in China mostly lies in the range of 1m to 100m and there are also some buildings covers large areas.

C. SPATIAL ERROR

Because we use a spatial grid to describe the location of object in the meshing system, it is inevitable to introduce some position error when using the AHG code to do spatial query. As descript in Coding Process part, spatial object with the MBR size of lat $\times$lon will be coded at the level of Ladapt under the restriction of following formula:

$$\frac{360}{2^{l_{adapt}+1}} < \max\{\Delta lon, 2 \times \Delta lat\} \leqslant \frac{360}{2^{l_{adapt}}}, \quad l_{adapt} \in \mathbb{N}^{+}$$

Assume $d_{obj} = max\{lon, 2 \times lat\}$, we can get the coding level by the following formula:

$$\log_2\left(360/d_{obj}\right) - 1 < l_{adapt} \leqslant \log_2\left(360/d_{obj}\right), \quad l_{adapt} \in \mathbb{N}^{+}$$

And the grid size of the coding level is $360/2^{l_{adapt}}$.

As mentioned above, when we apply AHG code to spatial neighbor query, we extend the query scope to the eight neighborhoods of the target point. So the query error is within 2 grids, that is $360/2^{l_{adapt}-1}$ rad.

For spatial size query, we use the length of the AHG code to filter the candidate object. The query error is within half of the grid size cause the hierarchical level is adapted to the size of spatial object and current coding level is the most accurate level to describe the location of the object. Namely, the spatial object's size lies between the current grid and the one deeper level grid. Therefore, the size query error is $360/2^{l_{adapt}+1}$ rad.

VI. CONCLUSION

Compared to traditional Geohash method, AHG can not only serve as a global meshing system for almost all kinds of spatial object, but also provides a useful pre-filtering tool for various spatial applications. AHG meshing and encoding method utilizes the spatial locality of Hilbert space-filling curve and uses an adaptive algorithm to adjust the coding length according to the size of the spatial object. The meshing hierarchical level can reflect the spatial size information and the coding result can express spatial location information, which contribute to a broader range of spatial applications than traditional Geohash code. Such as in spatial range and spatial neighbor query, AHG coding can be used to filter out most objects that do not meet the query criteria by a simple operation of string prefix matching. The method is now applied successfully in a high-performance geographic information system called HiGIS. It works effectively in accelerating the process of spatial range query and spatial neighbor query as Fig. 14 and Fig. 15 illustrate.

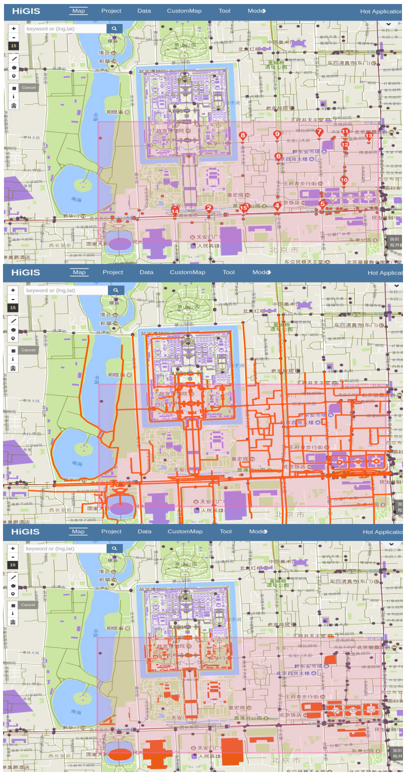


FIGURE 14. **Spatial range query of different datasets in HiGIS.**

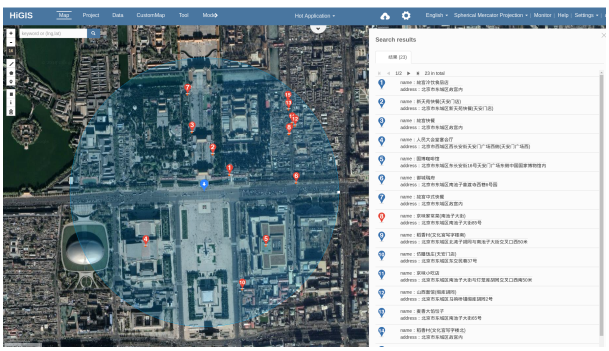


FIGURE 15. **Spatial neighbor query in HiGIS.**

Apart from this, AHG can also support a new type of spatial size query. It can directly screen out the object whose size is within a given interval without calculating the precise size. It brings great convenience to regional spatial analysis and helps to reveal some valuable information in a very short time, which is also meaningful for data mining of massive spatial dataset.

Though AHG meshing and coding method has a wide range of applications in practical, it is essentially a method based on geodetic coordinates. The closer to the Earth's poles, the worse the mesh fits the earth's surface, and the greater the distance distortion. It is a widely existing problem caused by the Mercator projection system. As a result, the mesh grids in high latitude areas are not exactly rectangular and the size of the grids represent a smaller distance compare to the grid near the equator. The phenomenon will bring nonnegligible error to spatial neighbor query and the size query if using AHG code. Therefore, in order to decrease the error of meshing for high latitude area, the mesh correction strategy and the corresponding coding method should be further researched.

ACKNOWLEDGMENT

The authors would like to thanks to the anonymous reviewers who gave me plenty valuable advices to improve the paper.

REFERENCES

[1] A. Mahdavi-Amiri, *A Survey of Digital Earth*, New York, NY, USA: Pergamon, 2015.
[2] C. Q. Cheng and C. Fu, "Earth space reference grid and its application prospect," *Geomatics World*, vol. 7, no. 3, pp. 1–8, 2014.
[3] K. Sahr, D. White, and A. J. Kimerling, "Geodesic discrete global grid systems," *Cartography Geograph. Inf. Sci.*, vol. 30, no. 2, pp. 121–134, 2003.
[4] Y. Jie-qing and W. U. Li-Xin, "Spatial subdivision and coding of a global three-dimensional grid: Spheoid Degenerated-Octree Grid," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2009, pp. II-361–II-364.
[5] G. M. Morton, "A computer oriented geodetic data base; And a new technique in file sequencing," *Phys. Plasmas*, vol. 24, no. 7, pp. 159–173, 1966.
[6] S. H. Song, C. Q. Cheng, G. L. Pu, F. G. An, and X. Luo, "Global remote sensing data subdivision organization based on GeoSOT," *Acta Geodaetica Cartographica Sinica*, vol. 43, no. 8, pp. 869–876, 2014.
[7] Z. Sun and C. Cheng, "True 3D data expression based on GeoSOT-3D ellipsoid subdivision," *Geomatics World*, vol. 23, no. 3, pp. 40–46, 2016.
[8] K. M. Górski *et al.*, "HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere," *Astrophys. J.*, vol. 622, no. 2, p. 759, 2005.
[9] M. Lambers and A. Kolb, "Ellipsoidal cube maps for accurate rendering of planetary-scale Terrain data," in *Proc. Pacific Conf. Comput. Graph. Appl. (PGP)*, 2012, pp. 5–10.
[10] T. Bernardin *et al.*, "Crusta: A new virtual globe for real-time visualization of sub-meter digital topography at planetary scales," *Comput. Geosci.*, vol. 37, no. 1, pp. 75–85, 2011.
[11] G. H. Dutton, *A Hierarchical Coordinate System for Geoprocessing and Cartography* (Lecture notes in earth sciences). Berlin, Germany: Springer, 1999.
[12] M. F. Goodchild and Y. Shiren, "A hierarchical spatial data structure for global geographic information systems," *CVGIP, Graph. Models Image Process.*, vol. 54, no. 1, pp. 31–44, 1992.
[13] A. Vince and X. Zheng, "Arithmetic and Fourier transform for the PYXIS multi-resolution digital Earth model," *Int. J. Digit. Earth*, vol. 2, no. 1, pp. 59–79, 2009.
[14] P. Peterson, "Close-packed, uniformly adjacent multiresolutional, overlapping spatial data ordering," U.S. Patent US8 400 451 B2, Mar. 19, 2013.
[15] K. Sahr, "Central place indexing systems," U.S. Patent 20 120 206 494 A1, Oct. 28, 2010.
[16] X. Tong, J. Ben, Y. Wang, Y. Zhang, and T. Pei, "Efficient encoding and spatial operation scheme for aperture 4 hexagonal discrete global grid system," *Int. J. Geograph. Inf. Sci.*, vol. 27, no. 5, pp. 898–921, 2013.
[17] X. Cao, G. Wan, F. Li, and K. Li, "The extended-octree spheroid subdivision and coding model," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 40, no. 4, pp. 95–100, 2013.
[18] J. X. Wang, Y. H. Li, Y. S. Zheng, and J. N. Liu, "Global 3D-grids based on great circle arc QTM sphere Octree and its application," *Geomatics Inf. Sci. Wuhan Univ.*, vol. 38, no. 3, pp. 344–348, 2013.
[19] T. Rees, "C-Squares, a new spatial indexing system and its applicability to the description of oceanographic datasets," *Oceanography*, vol. 16, no. 1, pp. 9–11, 2003.

[20] C. C. Tanner, C. J. Migdal, and M. T. Jones, "The clipmap: A virtual mipmap," in *Proc. 25th Annu. Conf. Comput. Graph. Interact. Techn.*, 1998, pp. 151–158.

[21] J. Yu, L. Wu, G. J. Zi, and Z. Guo, "SDOG-based multi-scale 3D modeling and visualization on global lithosphere," *Sci. China Earth Sci.*, vol. 55, no. 6, pp. 1012–1020, 2012.

[22] *FGDC-STD-011-2001, United States National Grid*, Standards Working Group, Federal Geographic Data Committee, Reston, VA, USA, 2001.

[23] *World Geographic Reference System (GEOREF), Van Nostrand's Scientific Encyclopedia*. Hoboken, NJ, USA: Wiley, 2005.

[24] *GB/T 12409-2009, Geographic Grid*, Nat. Geographic Inf. Standardization Tech. Committee, Beijing, China, 2009.

[25] R. Lampinen, "Universal transverse mercator (UTM) and military grid reference system (MGRS)," Nat. Geospatial-Intell. Agency, Reston, VA, USA, Tech. Rep., 2001.

[26] *STANAG 2577 ED.1, Nato Specifications for Global Area Reference System*, Nat. Geospatial-Intell. Agency, Reston, VA, USA, 2012.

[27] S. Zelinka, "Open location code," Google, Mountain View, CA, USA, Tech. Rep., 2015.

[28] L. H. Van and A. Takasu, "An efficient distributed index for geospatial databases," in *Proc. Int. Conf. Database Expert Syst. Appl.* Cham, Switzerland: Springer, 2015, pp. 28–42.

[29] M. Gao, L. Xiang, and J. Gong, "Organizing large-scale trajectories with adaptive Geohash-tree based on secondo database," in *Proc. 25th Int. Conf. Geoinf.*, Aug. 2017, pp. 1–6.

[30] L. Xiang, D. Wang, and J. Gong, "Organization and efficient range query of large trajectory data based on geohash," *Geomatics Inf. Sci. Wuhan Univ.*, vol. 42, no. 1, pp. 21–27, 2017.

[31] H. Jiang *et al.*, "Vector spatial big data storage and optimized query based on the multi-level Hilbert grid index in HBase," *Information*, vol. 9, no. 5, p. 116, 2018.

[32] T. Vukovic, "Hilbert-Geohash-Hashing geographical point data using the Hilbert space-filling curve," M.S. thesis, NTNU, Trondheim, Norway, 2016.

[33] S. Josefsson, *The Base16, Base32, and Base64 Data Encodings*, document RFC 4648, 2006.

[34] J. An, C. Cheng-Qi, S. Shu-Hua, and C. Bo, "Regional query of area data based on Geohash," *Geography Geo-Inf. Sci.*, vol. 29, no. 5, pp. 31–35, 2013.

[35] J. Liu, H. Li, Y. Gao, H. Yu, and D. Jiang, "A geohash-based index for spatial data management in distributed memory," in *Proc. 22nd Int. Conf. Geoinf.*, Kaohsiung, Taiwan, Jun. 2014, pp. 1–4.

[36] S. He, L. Chu, and X. Li, "Spatial query processing for location based application on Hbase," in *Proc. IEEE 2nd Int. Conf. Big Data Anal.*, Beijing, China, Mar. 2017, pp. 110–114.

[37] K. Lee, R. K. Ganti, M. Srivatsa, and L. Liu, "Efficient spatial query processing for big data," in *Proc. 22nd ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2014, pp. 4–7.

[38] K. Huang, G. Li, and J. Wang, "Rapid retrieval strategy for massive remote sensing metadata based on geohash coding," *Remote Sens. Lett.*, vol. 9, no. 11, pp. 1070–1078, 2018.

[39] R. Moussalli, M. Srivatsa, and S. Asaad, "Fast and flexible conversion of geohash codes to and from latitude/longitude coordinates," in *Proc. IEEE 23rd Annu. Int. Symp. Field-Program. Custom Comput. Mach.*, Vancouver, BC, USA, May 2015, pp. 179–186.

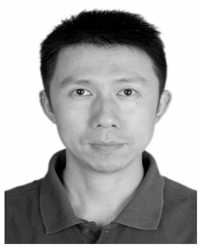
[40] Y.-K. Huang, "Location-based aggregate queries for heterogeneous neighboring objects," *IEEE Access*, vol. 5, pp. 4887–4899, 2017.

[41] Y. Cao *et al.*, "Binary hashing for approximate nearest neighbor search on big data: A survey," *IEEE Access*, vol. 6, pp. 2039–2054, 2017.

NING GUO was born in Jiangsu, China, in 1992. He received the B.S. degree in information engineering and the M.S. degree in information and communication engineering from the National University of Defense Technology, Changsha, China, in 2014 and 2016, respectively, where he is currently pursuing the Ph.D. degree with the Spatial Information System Laboratory. His research interests include GIS, spatial database, and spatio-temporal data modeling and processing.

WEI XIONG was born in Hunan, China. He received the M.S. degree in computer engineering from the Naval University of Engineering, Wuhan, China, in 2001, and the Ph.D. degree in information and communication engineering from the National University of Defense Technology, Changsha, China, in 2005, respectively, where he is currently an Associate Professor. His research interests include geographic information systems and spatial database.

YE WU was born in Hunan, China. He received the M.S. degree in information engineering and the Ph.D. degree in information and communication engineering from the National University of Defense Technology, Changsha, China, in 2008 and 2015, respectively, where he is currently a Lecturer. His research interests include geographic information systems and spatial database.



LUO CHEN received the bachelor's degree in signal and information processing and the Ph.D. degree in information and communication engineering from the National University of Defense Technology, in 1994 and 2005, respectively. He has been a Professor of photogrammetric and remote sensing with the National University of Defense Technology, since 2010. He has authored many peer-reviewed papers in journals and conferences. His research and teaching interests include geospatial information processing and systems (GIS), spatial analysis and modeling, parallel and distributed computing. He is a member of ACM and China Computer Federation (CCF). He is the Principal Investigator (PI) of several projects supported by the High Technology Research and Development Projects of China, and was a PI and CI of projects supported by the National Science Foundation of China.

NING JING received the B.Sc. degree in communication and information systems, the M.Sc. degree in signal and information processing, and the Ph.D. degree in computer science from the National University of Defense Technology, in 1983, 1986, and 1990, respectively. He has been served as an Expert in earth observation and navigation area of the National High-Tech R&D Program of China. He is currently a Professor and the Director of the Department of Information Engineering, National University of Defense Technology. His research interests include geographical information systems, database systems, planning and decision support in spatial resources, spatial data analysis, and visualization. He is a Senior Fellow of the China Computer Federation (CCF), a Fellow of the Technical Committee of Database System of CCF, a Vice Director of the Technical Committee of Public Security, and a Fellow of the Technical Committee of Principles and Methods of the China GIS Association.