International Journal of
Geo-Information

MDPI

Article

An On-Demand Retrieval Method Based on Hybrid NoSQL for Multi-Layer Image Tiles in Disaster Reduction Visualization

Linyao Qiu [1,2], Qing Zhu [1,2,3,4], Zhiqiang Du [1,2,*], Meng Wang [1,2] and Yida Fan [5]

1 State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing (LIESMARS), Wuhan University, 129 Luoyu Road, Wuhan 430079, China; qiu_linyao@163.com (L.Q.); zhuq66@263.net (Q.Z.); eclipse_sv@163.com (M.W.)
2 Collaborative Innovation Centre for Geospatial Technology, Wuhan University, 129 Luoyu Road, Wuhan 430079, China
3 Faculty of Geosciences and Environmental Engineering of Southwest Jiaotong University, 999 Xi'an Road, Chengdu 611756, China
4 State-Province Joint Engineering Laboratory of Spatial Information Technology for High-Speed Railway Safety, 999 Xi'an Road, Chengdu 611756, China
5 National Disaster Reduction Centre of China, Ministry of Civil Affairs, 6 Guangbai East Road, Beijing 100124, China; fanyida@126.com
* Correspondence: duzhiqiang@whu.edu.cn; Tel.: +86-27-6877-8779

Academic Editor: Wolfgang Kainz
Received: 20 November 2016; Accepted: 5 January 2017; Published: 9 January 2017

Abstract: Monitoring, response, mitigation and damage assessment of disasters places a wide variety of demands on the spatial and temporal resolutions of remote sensing images. Images are divided into tile pyramids by data sources or resolutions and published as independent image services for visualization. A disaster-affected area is commonly covered by multiple image layers to express hierarchical surface information, which generates a large amount of namesake tiles from different layers that overlay the same location. The traditional tile retrieval method for visualization cannot distinguish between distinct layers and traverses all image datasets for each tile query. This process produces redundant queries and invalid access that can seriously affect the visualization performance of clients, servers and network transmission. This paper proposes an on-demand retrieval method for multi-layer images and defines semantic annotations to enrich the description of each dataset. By matching visualization demands with the semantic information of datasets, this method automatically filters inappropriate layers and finds the most suitable layer for the final tile query. The design and implementation are based on a two-layer NoSQL database architecture that provides scheduling optimization and concurrent processing capability. The experimental results reflect the effectiveness and stability of the approach for multi-layer retrieval in disaster reduction visualization.

Keywords: on-demand; multi-layer; semantic description; NoSQL; disaster reduction visualization

1. Introduction

Rapid advancements in Earth-observing systems have led to a large amount of remote sensing data that can be used in disaster monitoring, response, mitigation and recovery [1–3]. These data have become significant in the application of geographic information technology to disaster reduction. Additionally, such systems can be used to retrieve high-spatiotemporal resolution images and establish global coverage of digital earth through aerospace remote sensing technology [4–7].

Remote sensing images play a critical role in the disaster reduction process because they deliver spatially related information in a direct and intuitive manner [8]. Based on the discrete global grid system [9], observed images are commonly cut into tiles and incorporated into pyramid models to provide broad data service for every phase of disaster management. For instance, high-resolution image layers are widely used as base maps for collaborative plotting [10] and feature extraction [1] in disaster mitigation processes [11].

An image layer is a complete or partially continuous tile pyramid that is built from a set of images. Tiles of different levels or areas of the pyramid can be cut from various image sources. Using Google Maps as an example, its tiles originate from QUICKBIRD, LANDSAT, IKONOS and other satellite sensors [12,13]. Each tile in a specific layer is unique, with its coding representing a single three-dimensional spatial location on the Earth's surface.

In the disaster reduction process, the demand for image layers is diverse and ever changing. Using disaster warning as an example, the monitoring of different disasters is associated with distinct requirements regarding the spatial and temporal resolutions of observed remote sensing images (see Figure 1). Moreover, different tasks in one disaster reduction process introduce different needs regarding temporal scales and image resolution [14]. In a complete disaster reduction visualization, the disaster scene continues to access various image tiles that differ with respect to their temporal and spatial resolutions to satisfy different degrees of terrain feature recognition. For instance, in the process of flood reduction, the basic disaster environmental visualization requires 1 km spatial resolution for daily monitoring [15]. Additionally, 5 m to 10 m spatial resolution is required for water extraction in the response phase [1], and sub-meter resolution is required for the extraction of flood-affected buildings in the loss assessment phase [16]. These images are generally continuously accessed during each phase of disaster reduction; thus, it is difficult to pre-organize them into the same pyramid-based dataset. Therefore, many datasets are independently stored for diverse disaster reduction tasks. When a disaster occurs, real-time observed images are obtained and used to build new pyramids. Then, new image layers are released to represent the destroyed areas with different spatial and temporal resolutions, in addition to the historical image layers.

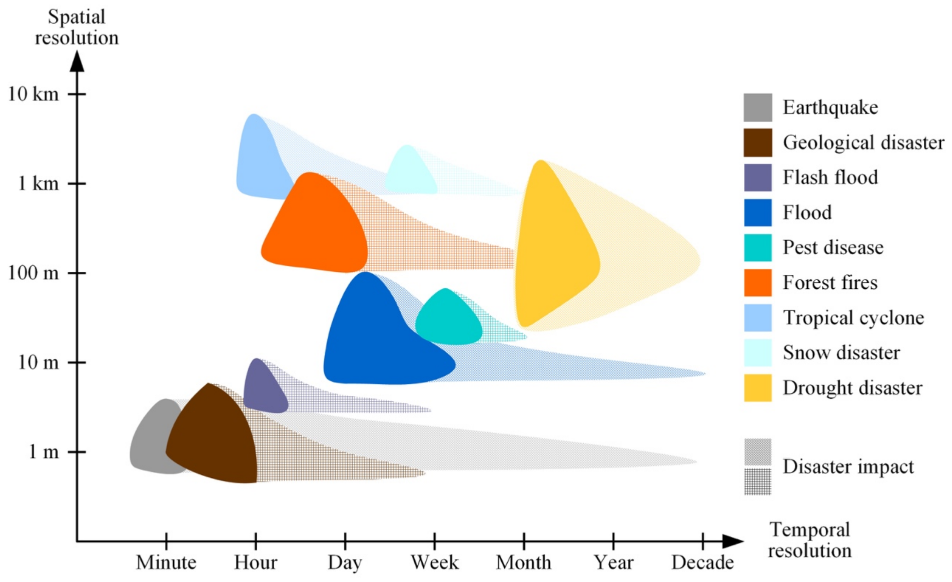


Figure 1. Spatial and temporal resolution requirements of remote sensing images in disaster reduction (taken from "Disaster Reduction Application System Feasibility Analysis Report of Small Satellite Constellation for Monitoring and Forecasting of Environment and Disaster, 2006, National Disaster Reduction Committee of China").

With the advancement in disaster reduction studies, an increasingly large number of remote sensing image services overlap in disaster-affected areas. Therefore, tiles that have the same name but are from different datasets will create redundancies in the process of tile retrieval, transmission and visualization. This paper proposes a hybrid NoSQL-based on-demand retrieval method to address this inefficiency in multi-layer image tile services. This method is twofold. First, it provides a layer description model based on semantics, which is used to represent more dataset information and build correlations based on task demands. In the retrieval process, the method automatically filters irrelevant layers and selects the most suitable layers for tile retrieval by matching dataset semantic information with the real-time visualization demand. The second objective is the implementation of the description model and tile selection process based on a two-layer NoSQL database architecture. An in-memory distributed database is used as the first layer for tile caching, and a document database is used as the second layer for the effective storage and querying of many tiles. Moreover, at the transmission level, the HTTP/2.0 protocol is adopted to promote tile scheduling efficiency.

The remainder of this paper is organized as follows. Section 2 introduces the concept of multi-layer pyramid overlapping and the visualization requirements. In Section 3, the semantic model and on-demand tile matching flow are presented, while Section 4 introduces the two-layer database design and its implementation. Section 5 presents the experiment. A performance evaluation and the associated discussion are presented in Section 6. Finally, Section 7 offers conclusions.

2. Background and Motivation

The global multi-resolution pyramid is a discrete global grid model that defines a hierarchical division of the surface of the Earth [17,18]. Many division methods have been proposed in the literature, such as the spherical quad-tree [19], ISEA model, QTM and latitude/longitude model. The latitude/longitude model is the most practical division model in actual applications because the structure is easily understandable and operable for users to manage large-scale and multi-resolution images and to locate any fixed grid space by applying the following division rules [20]:

1. The surface of the Earth is transformed to a regular rectangle ranging from −180 to 180 degrees in longitude and −90 to 90 degrees in latitude. Values outside of this area are invalid.
2. The tile span at pyramid level k is two times that at level k + 1. The global pyramid starts from level 0, which has two tiles whose spans are each 180 degrees.
3. The ratio of the number of pyramid tiles between the transversal and vertical directions is 2:1 at any pyramid level.
4. The coding order for pyramid tiles starts from west to east and from south to north at any pyramid level.
5. The coding order for pyramid levels starts from top to bottom.
6. When the geographical location and height are specified globally, the resulting tile is unique.

Based on the rules, any image with a geospatial reference can be mapped into the global grid model. Furthermore, the image can be cut into an independent local pyramid (see Figure 2). The maximum level of the local pyramid is determined based on the spatial resolution of the image. Suppose the pixel size of the image is $s \times t$; then, the spatial resolution of tiles ($f(l)$) in level l can be calculated (taking the longitude direction as an example) as follows.

$$f_x(l) = 180.0/\left(2^l \times \mathrm{s}\right) \tag{1}$$

Suppose the spatial area of the image is A and its pixel matrix is $m \times n$; then, the spatial resolution of the image ($r(x)$) can be defined (taking the longitudinal direction as an example) as follows:

$$r(x) = (A.x_h - A.x_t)/m; r(y) = (A.y_h - A.y_t)/m \tag{2}$$

where (x_t, y_t) and (x_h, y_h) are the minimum and maximum coordinates of area A, respectively. Therefore, the maximum level of the local pyramid (l_{max}) can be calculated based on Equations (1) and (2).

$$l_{max} = max\left\{ \left| log_2 \frac{180.0}{r(x) \times s} \right|, \left| log_2 \frac{180.0}{r(y) \times t} \right| \right\} \tag{3}$$

Thus, a unique tile can be computed based on the level, column and row number. Moreover, if a longitude and latitude (x, y) are given, the associated tile coding (X, Y) in any level of the pyramid can be retrieved.

$$X = \left| \frac{x + 180}{f_x(l)} \right|, Y = \left| \frac{y + 180}{f_y(l)} \right| \tag{4}$$

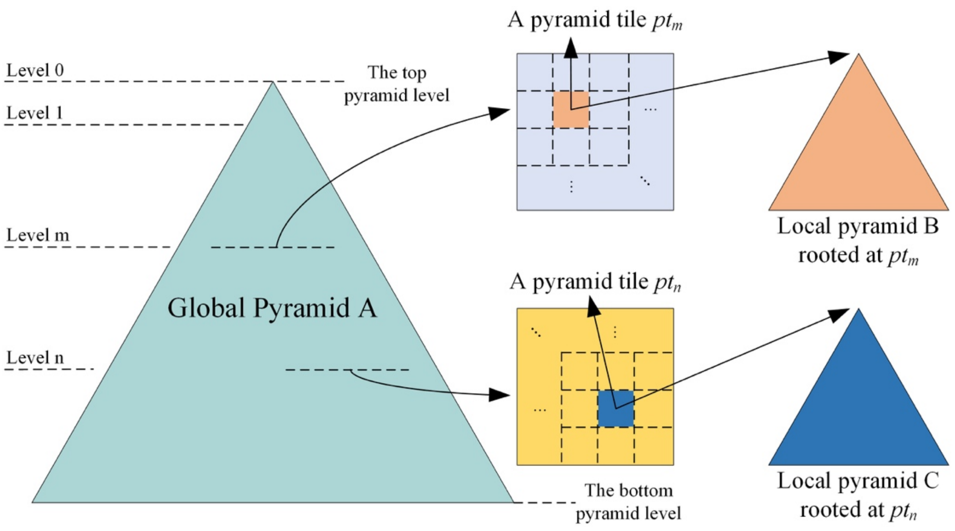


Figure 2. The relationship between the global pyramid and local pyramids.

Clearly, when the pyramid layer is single, tile visualization is simple because the result of a retrieval based on the above formulas is unique. However, in disaster reduction applications, multiple pyramids are produced and overlaid in the same disaster-affected area. Every new layer is an independent sub-pyramid from a respective data source. As shown in Figure 3, layers A, B and C overlap at pyramids of level k, k + 1 and k + 2 in the same area, which produces namesake tiles that belong to the respective layers at these levels. When viewpoint retrieval is used for a tile in this region, it cannot distinguish the namesake tiles from these layers. Repeated tile access and loading lead to visual confusion and discontinuity in refreshed scenes, and more resources are wasted in servers and transmission.

The traditional tile retrieval method is oriented toward a single layer, and it calculates a unique tile code based on the spatial location of the current viewpoint [21,22]. When multiple image layers are overlaid in a spatial region, redundant tiles can exist with the same name but from different layers in certain areas that correspond to the spatial intersection of multiple image datasets [23]. Because the viewpoint-based tile retrieval method cannot distinguish the target tile from namesake tiles as multiple layers are overlaid, users generally perform layer switching to ensure that the visible layer is unique at any given time [24]. In such a situation, the target layer is clear before any tile searching occurs.

However, in a disaster reduction process, real-time image data are continuously accessed to build new layers, each of which is considerably different with respect to the temporal and spatial resolutions and extents [25]. Loading only one layer at a time can neither represent the complete disaster characteristics of a disaster-affected area nor satisfy the variety of spatial-temporal resolution demands in the process of disaster reduction visualization. Additionally, it is complex and time consuming to integrate various datasets with different spatial resolutions and scale them into one pyramid, which may also lead to temporal chaos if the temporal resolutions of the integrated datasets do not match.

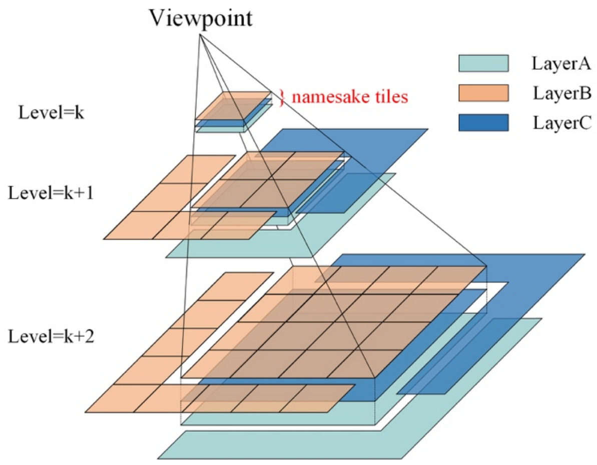


Figure 3. Overlapping of multiple image layers.

The viewpoint-based tile retrieval method cannot filter the namesake tiles from multiple layers; it must traverse every dataset stored in the database according to tile level, row and column codes [26]. Then, all of the searched tiles are repeatedly loaded from the server to the client. Clearly, considerable tile redundancy occurs and wastes service resources in the process of retrieval. In the servers, every tile request forces the database to search all of the image datasets and return a group of namesake results, causing repeated and unnecessary retrieval in irrelevant datasets, as well as query speed degradation. In the clients, the namesake tiles are received, parsed and drawn for visualization individually. This process considerably decreases the refresh speed of the visual scene and greatly increases the memory pressure of the clients. In the transfer layer, large numbers of invalid tiles occupy the majority of the bandwidth resources, which seriously affects the transmission rate of available tiles. Moreover, as the number of datasets stored in the database increases, the wasted resources and low efficiency of retrieval become more serious.

An on-demand retrieval method for multi-layer image tiles is proposed in this paper. It forces servers to automatically analyze the visualization demand based on the viewpoint location and reduction task information. Then, the tile is actively selected from suitable layers to respond to the clients, instead of traversing all the datasets or relying on a man-machine layer control switch. As shown in Figure 4, when the viewpoint is far from the surface of the Earth, servers provide tiles from the low-resolution pyramid A for basic visualization. As the viewpoint moves to the flood-affected area, the servers select medium-resolution pyramid B for the flood extraction tasks. When the viewpoint is close to the surface of the flooded city, the servers actively choose high-resolution tiles only from pyramid C for the loss assessment tasks. This autonomy takes full advantage of the flexibility and diversity of the multi-layer image service to satisfy the various demands of disaster reduction visualization and guarantee an efficient tile request for 3D scene refreshing by the clients.

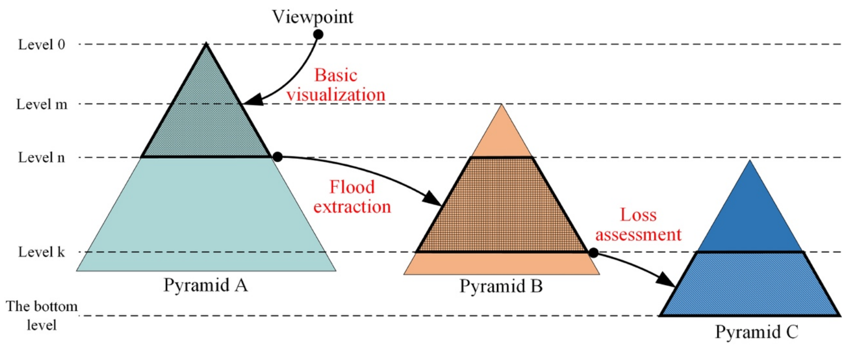


Figure 4. Diverse tile requirements in the 3D visualization of a disaster area.

3. Methodology

The viewpoint-based method can easily retrieve unique tiles from a single layer, but as the number of overlaid layers increases, this method cannot distinguish the most suitable tiles from multiple datasets. To overcome this problem, we propose a semantic description model of image layers. Multi-dimensional semantics are defined in the model, and an annotation approach based on the resource description framework (RDF) is used to describe the layers. Layer selection and tile filtering are performed in a matching process.

3.1. Semantic Description of an Image Layer

Semantic annotation has been widely used in geoprocessing, particularly in the composition of semantic web services [27,28]. This approach features high semantic integrity and formalizes metadata representation to make the metadata machine readable [29], which improve intercomprehension and interoperation among datasets [30,31].

The semantic description model contains five elements: the spatial semantics describe the spatial region range of a dataset; temporal semantics record the collection time of remote sensing images; resolution semantics define the vertical range of the local pyramid; priority semantics control the order in which datasets are scheduled; and theme semantics are a set of sub-tags that record disaster information, such as disaster areas, disaster types, and other factors. The detailed descriptions and functions of the semantics are as follows.

3.1.1. Spatial Semantics

The two-dimensional range of a dataset on the surface of the Earth is described by spatial semantics. In the disaster reduction process, a new set of images is typically cut and joined into one image whose shape seamlessly fits with the administrative region of the disaster area. Spatial semantics represent a complex polygon shape that accurately describes the true extent of the composite image. Spatial semantics are used to precisely filter irrelevant datasets that are outside the viewport.

3.1.2. Temporal Semantics

Temporal semantics describe the order and life cycle of the dataset in the time dimension. The time range of each dataset is defined in the form of a time stamp that provides support for sequence analysis and multi-temporal presentation of multiple layers. These are typical applications in which tiles from multiple layers are requested and scheduled at the same time; thus, the stamp is the key indicator that controls the selection sequence of namesake tiles.

3.1.3. Resolution Semantics

The spatial resolution of an image represents the maximum level of its pyramid. In the pyramid building process, the maximum level of the current pyramid reflects the resolution semantics. It specifies the visible extent of the image layers in the vertical range of 3D space, which represents the functional information of the current pyramid. For instance, when the viewpoint reaches a location that is close to the surface, it implies that the user is more concerned with the feature details of the high-resolution image dataset than the low-resolution texture of the global background dataset. In such a situation, the level information is applied to further exclude layers that are outside the field of view of interest.

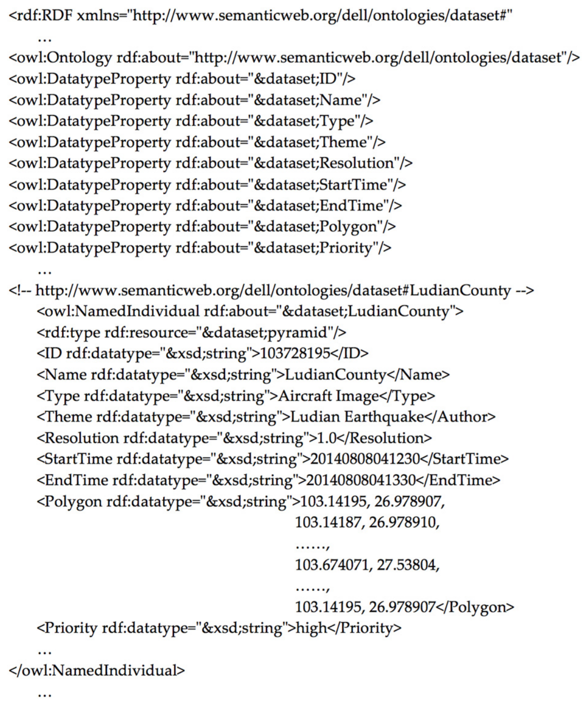
3.1.4. Priority Semantics

Priority is the secondary index that determines the order in which the layers are loaded. In the emergency response and loss assessment stages of disaster reduction, there are many new layers that overlap with respect to the spatial extent, visible depth and life cycle. Priority is used to distinguish the highly overlapping layers. The priority is initialized by artificial experience as the pyramid is stored in the database. For example, if current images are more suitable for water extraction in a flood disaster, the priority of the corresponding pyramid is higher than that of pyramids from other sensors. A default setting is supported, as the latest stored pyramid has a higher priority when other indicators are the same. This setting is used to ensure that the tile search process can always find the only appropriate layer to be distinguished from other layers based on the tile uniqueness rule of global discrete grid division.

3.1.5. Theme Semantics

The theme semantic constraint is established to distinguish image differences in terms of sensor types, data sources, related disaster events and other information. In the disaster area visualization task, different disasters feature various preferences regarding image resolution and corresponding sensor types. In theme semantics, a group of sub-tags is built to describe the theme characteristics of the image layer. It also effectively increases the association constraints of different visualization tasks and datasets, which further supports users in selecting and discarding layers in batches.

The RDF is adopted to represent the semantic annotation. An RDF file effectively organizes the semantics of each dataset and transforms them into resource information that the database can easily identify and analyze. An RDF file, as shown in Scheme 1, is produced when a new pyramid has been built. Then, the annotation information is interpreted and stored in the database for the matching process.



```
<rdf:RDF xmlns="http://www.semanticweb.org/dell/ontologies/dataset#"
    …
<owl:Ontology rdf:about="http://www.semanticweb.org/dell/ontologies/dataset"/>
<owl:DatatypeProperty rdf:about="&dataset;ID"/>
<owl:DatatypeProperty rdf:about="&dataset;Name"/>
<owl:DatatypeProperty rdf:about="&dataset;Type"/>
<owl:DatatypeProperty rdf:about="&dataset;Theme"/>
<owl:DatatypeProperty rdf:about="&dataset;Resolution"/>
<owl:DatatypeProperty rdf:about="&dataset;StartTime"/>
<owl:DatatypeProperty rdf:about="&dataset;EndTime"/>
<owl:DatatypeProperty rdf:about="&dataset;Polygon"/>
<owl:DatatypeProperty rdf:about="&dataset;Priority"/>
    …
<!-- http://www.semanticweb.org/dell/ontologies/dataset#LudianCounty -->
    <owl:NamedIndividual rdf:about="&dataset;LudianCounty">
    <rdf:type rdf:resource="&dataset;pyramid"/>
    <ID rdf:datatype="&xsd;string">103728195</ID>
    <Name rdf:datatype="&xsd;string">LudianCounty</Name>
    <Type rdf:datatype="&xsd;string">Aircraft Image</Type>
    <Theme rdf:datatype="&xsd;string">Ludian Earthquake</Author>
    <Resolution rdf:datatype="&xsd;string">1.0</Resolution>
    <StartTime rdf:datatype="&xsd;string">20140808041230</StartTime>
    <EndTime rdf:datatype="&xsd;string">20140808041330</EndTime>
    <Polygon rdf:datatype="&xsd;string">103.14195, 26.978907,
                                        103.14187, 26.978910,
                                        ……,
                                        103.674071, 27.53804,
                                        ……,
                                        103.14195, 26.978907</Polygon>
    <Priority rdf:datatype="&xsd;string">high</Priority>
    …
</owl:NamedIndividual>
    …
```

Scheme 1. Selected parts of a dataset description in an RDF file.

3.2. *The Matching Process*

The traditional viewpoint-based tile retrieval method cannot determine the differences among namesake tiles belonging to different layers; instead, it must traverse all the datasets to find available tiles. The improvement presented in this paper lies in the description of the multi-dimensional features of the dataset based on semantic information and adaptive matching tiles combined with the current viewpoint position information in the retrieval process. Figure 5 shows the process of automatic tile matching and filtering.

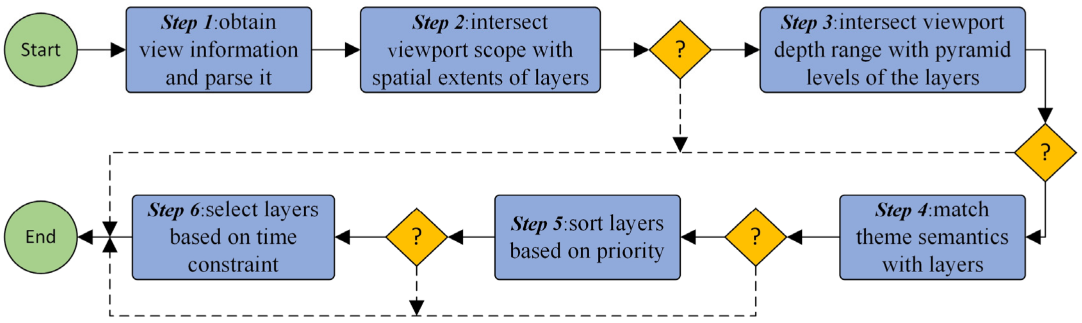

Start
Step 1:obtain view information and parse it
Step 2:intersect viewport scope with spatial extents of layers
?
Step 3:intersect viewport depth range with pyramid levels of the layers
?
Step 4:match theme semantics with layers
?
Step 5:sort layers based on priority
?
Step 6:select layers based on time constraint
End

Figure 5. On-demand tile matching and filtering process.

The detailed steps of the above process are as follows:

***Step* 1** A tile request is obtained and first parsed into requirement semantics of visualization tasks, including the geographic extent of the viewport, pyramid level of the tile, theme information, priority requirement and time constraints.

***Step* 2** The metadata of all the datasets are traversed to perform the intersection operation based on the viewport range and boundary polygon of every dataset to find datasets *list1* in which every member intersects the viewport.

***Step* 3** Traverse *list1* to select datasets whose pyramid level range contains the current requested level; the search results generate *list2*.

***Step* 4** Match the theme information with the theme tabs of every dataset in *list2*; the matched datasets compose *list3*. Datasets in *list3* represent task preferences for layer services.

***Step* 5** In *list3*, sort datasets based on priority and select the dataset with the highest priority to generate *list4*.

***Step* 6** A time constraint is applied to select the most suitable dataset from *list4*. The default setting is to choose the latest dataset; a unique dataset could also be searched for by setting a special time period in the tile request condition.

Additionally, as shown in Figure 5, each diamond represents a judgment, such that if the current list has only one member, the ideal dataset has been selected, and the process goes straight to the end node. If the current list has no member, the process must lower the filter conditions by re-locating the previous list and skipping from the last filtering node to the next one. Specifically, if the first judgment node returns null, which means there are no suitable datasets intersecting the current viewport range, the process must stop and return null to the application server.

The process of layer matching and filtering can effectively select the most suitable image layer that satisfies the current visualization task. To take full advantage of server efficiency for data searching and information publishing, the implementation of the matching and filtering process is arranged on the server side. A two-layer NoSQL database architecture is designed to effectively meet the storage and access needs of mass datasets.

4. Implementation

A two-layer hybrid architecture based on key-value and document NoSQL databases is designed for efficient layer storage and tile retrieval. NoSQL databases have been widely used to storage and manage large amounts of image data retrieved via pyramid and hash indexing techniques on the server side [32]. The pyramid provides the rules of image cutting, and hash indexing gives a solution to store and search these tiles in a highly effective manner. NoSQL is one of the key technologies for large-capacity storage and fast retrieval oriented toward the continuous growth of datasets [33–36]. The main features of NoSQL databases are as follows [7,37]:

- Support of massively concurrent read and write operations and eventual consistency;
- Support of mass storage;
- Easy to expand;
- Low cost;
- Flexible data model that is non-structured or semi-structured; and
- Horizontal scalability.

The most popular categories of NoSQL databases include key-value databases and document databases. Key-value databases are used for fast and simple operations because they provide simple mapping from the key to the corresponding value, which yields fast object searches. Document databases offer flexible data models with query possibilities. They are considered an evolution of key-value databases because they include all the benefits of key-value databases while supporting strong and rich query capabilities.

4.1. Storage Design

The two-layer database architecture includes the storage layer and cache layer. The storage layer adopts MongoDB to manage large amounts of image layers and metadata. MongoDB provides a rich data structure to store the semantic information of datasets. In MongoDB, the GridFS structure supports efficient storage of mass tile data, and the collection structure provides effective metadata management [38]. Each tile pyramid is stored in GridFS, and the RDF file of its metadata is stored as a document in the metadata collection.

The cache layer provides an extensible cache pool based on Redis, a key-value NoSQL database. As an in-memory database, Redis has a high ability with respect to data reading, writing and querying tasks. Additionally, its horizontal scalability and distribution design support fast caching and the secondary access requirements of massive tile data. Cached tiles are evenly distributed among the memory nodes. The key of each tile is designed as "Dataset name: Tile location"; the dataset name is based on the GridFS name of datasets that contain the tile in MongoDB, and the tile location includes the row, column and level number of the discrete global grid system. This design ensures consistent tile indexes in the two databases.

On top of the two-layer databases, an application server is built to implement the layer matching process. When the tile request is received in the application server, the server parses the request and performs *step2* and *step3* (see Scheme 2) of the process via a spatial union query in MongoDB. The query results, including theme, time and priority semantics, return to the application server. The remaining steps of the process are quickly implemented in the server memory. The suitable layer is selected at the end of the matching process. Then, the server assembles the tile key for search in Redis. If no tile is found in the cache, then the sever searches again in MongoDB; the result is first returned to the server and subsequently sent to Redis for caching. The entire query process is shown in Figure 6.

```
# Joint query containing geospatial scope and level limit, returns a geojson object
result = col.find({
    "loc": {"$geoIntersects": {"$geometry": {
            "type": "Polygon",
            "coordinates": [
                [[90, 33], [110, 33], [110, 13], [90, 13], [90, 33]]
            ]
            }
    }},
    "max": {"$gt": 11},
    "min": {"$lt": 2}})
```

Scheme 2. Joint querying by combining the spatial scope and level constraint in MongoDB.

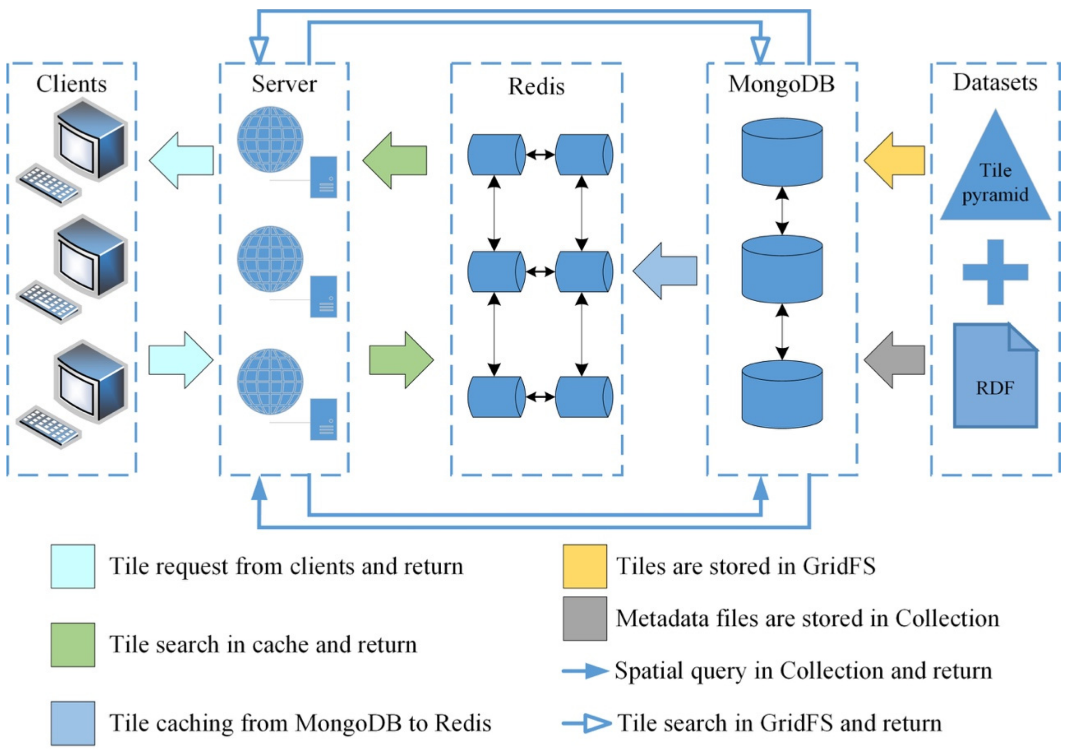

Clients
Server
Redis
MongoDB
Datasets
Tile pyramid
RDF
Tile request from clients and return
Tile search in cache and return
Tile caching from MongoDB to Redis
Tiles are stored in GridFS
Metadata files are stored in Collection
Spatial query in Collection and return
Tile search in GridFS and return

Figure 6. The entire tile request process from the clients and the response in the servers.

4.2. Scheduling Optimization

To enhance the response efficiency and transfer stability between the clients and servers, tile scheduling optimization is implemented, including the multi-queue least recently used (LRU) replacement method and HTTP/2.0-based multiplexing tile transmission.

4.2.1. Multi-Queue LRU Replacement

The technique of geospatial data caching is widely used in cloud-based environments [39,40]. A pyramid model of tiles provides a good management and caching method for geospatial data in a cloud-based environment [41]. Prefetching methods that predict data access [42,43] have been proposed to improve tile caching when the cache storage size is limited. As large memory size becomes more available in caching techniques, taking advantage of memory database abilities (large storage size and high-efficiency data retrieval and memory updating) could further enhance the tile hit rate and secondary access efficiency in the visualization process.

The LRU algorithm eliminates data according to the historical records of data access; its core idea is that if the data have been recently visited, then the probability of future visits is also higher [44], which is widely used in cache data management. The multi-queue LRU replacement algorithm divides the LRU queue into several queues, each with different access priorities. Multi-queue LRU has a higher hit rate than the traditional LRU algorithm but is also associated with higher complexity and computational costs. Because the clustered in-memory database has a sufficient memory capacity and query efficiency, the multi-queue LRU replacement algorithm is adopted to manage cached tiles in the memory.

As demonstrated in Figure 7, there are several queues in the cache, from *Q*0 to *Qk*, and the access priorities are addressed in turn. A new tile is stored in *Q*0 when it is requested by the client the first time. As the number of accesses increases beyond the threshold, the tile is removed from the current queue and added to the head of a high-level queue. If the tile has been not accessed for a certain time, it must be removed from the current queue and sent to the head of a low-level queue. Tiles at the end of any queue are removed and pushed to the head of a historical queue if its queue has been filled. The order of tile retrieval in the cache depends on the access priority of queues, and the history queue has the lowest priority. If a tile in the history queue is re-accessed, its priority is restored, and it goes back to the head of the last queue. Finally, the history queue clears the "useless" tiles according to the LRU rule. This algorithm makes full use of the efficient retrieval capability of Redis and takes advantage of the clustered memory space to make the storage time of popular tiles as long as possible, which ensures a high tile hit rate and improves the secondary access efficiency.

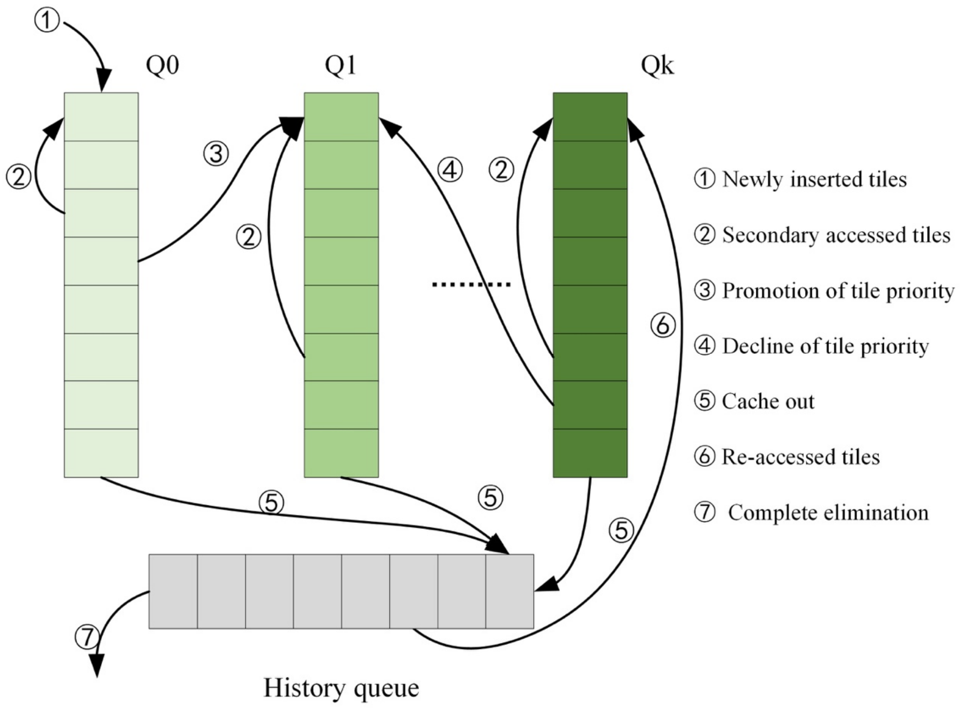


Figure 7. Multi-queue LRU replacement process.

4.2.2. HTTP/2.0-Based Multiplexing Tile Transmission

Tile requests from distributed clients are converted to HTTP requests and sent to the server. Then, the server parses the URL and searches the database. After the tiles are retrieved, they are returned to the clients through the network. HTTP is an application protocol that is most commonly used on the Internet; it is also the common language between the clients and servers.

Although most hypertext transfer protocols are based on HTTP/1.x, traditional HTTP/1.x transmission protocol-based data scheduling suffers from poor temporal efficiency and continuity. Repeated establishment and disconnection between the clients and servers leads to substantial time consumption, and the repeated transmission of the HTTP header results in a large amount of wasted network resources.

In a real-time tile visualization application, the application server must respond to many tile data requests in a relatively short period to ensure continuous visualization by the clients. In HTTP/1.x, a connection is established for a single request-response process, and the connection is closed when the response is received. The process of establishment and disconnection requires a large amount of extra transfer time. Moreover, HTTP/1.x uses the pipeline process approach, which queues several requests into a serial single-thread process. The request at the back of the queue must wait until the former request has been addressed. The single-thread method is prone to tile blocks and response timeouts when the number of requested tiles rapidly increases, which easily causes visualization blocking in the clients.

The HTTP/2.0-based transport protocol can achieve asynchronous connection multiplexing, which can greatly enhance the efficiency of tile transmission. In contrast to HTTP/1.x, which builds connections for every request-response, HTTP/2.0 establishes a continuous connection for all data requests. It decomposes a request message into multiple frames for transmission. The frames are reassembled at the receiving end so that a number of requests can be transmitted and received in parallel without affecting each other. The responses are also sent and received in this same way (see Figure 8). The multiplexing tile transmission of HTTP/2.0 fully uses the network to minimize tile blocking in the concurrent access situation.

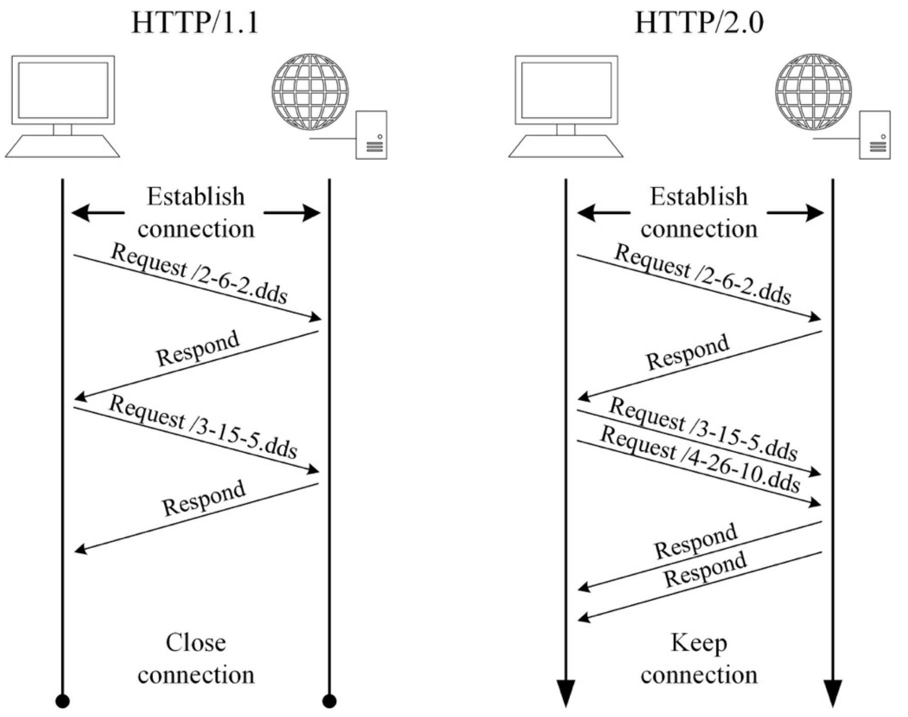


Figure 8. HTTP/2.0 multiplexing tile transmission process.

5. Experiment

5.1. Experimental Environment

Four Dell Edge Power R710 servers were virtualized into twelve virtual machines (VM) to establish a cluster-shared service resource. Among them, six VMs (1 core CPU running at 1.2 GHz and 8 GB of RAM) were used to build tile storage clusters and deploy the distributed document NoSQL database. Each VM was loaded with more than 1.5 TB of disk storage space (total of approximately 10 TB). The remaining six VMs (1 core CPU running at 1.2 GHz and 40 GB of RAM) were used to build cache clusters by deploying the key-value memory database cluster. Each VM was additionally allocated 40 GB of memory (total of 240 GB).

5.2. Experimental Dataset

The database stores more than 30 remote sensing image tile datasets related to two major earthquake disasters that caused substantial damage in China in recent years: the Lushan earthquake (30.3° N, 103.0° E), which occurred on 20 April 2014, and the Ludian earthquake (27.2° N, 103.4° E), which occurred on 3 August 2014. The datasets contain all the provincial 5 m resolution images of China and higher resolution satellite, aerial and UAV remote sensing images of cities, counties and villages in the disaster-affected provinces of China. The total dataset size is more than 5 TB. Table 1 presents the details of the experimental data.

Table 1. Details of experimental data.

Theme	Area	Spatial Resolution (m)	Source	Number of Datasets	Dataset Size (Gigabytes)
National basic images	34 provinces	5.00	Satellite image	1	4863.00
Ludian earthquake	Zhaotong City	2.50	Satellite image	1	150.00
	Ludian County	1.00	Satellite/aerial image	1	58.00
	Huize County	1.00	Satellite/aerial image	1	49.00
	Qiaojia County	1.00	Satellite/aerial image	1	73.00
	Zhaoyang County	1.00	Satellite/aerial image	1	55.00
	Local rural towns	0.20	Aerial/UAV image	18	380.00
Lushan earthquake	Yaan City	2.50	Satellite image	1	143.00
	Lushan County	1.00	Satellite/aerial image	1	33.00
	Baoxing County	1.00	Satellite/aerial image	1	63.00
	Tianquan County	1.00	Satellite/aerial image	1	58.00
	Nancheng County	1.00	Satellite/aerial image	1	48.00
	Minshan County	1.00	Satellite/aerial image	1	43.00
	Local rural towns	0.02	Aerial/UAV image	22	470.00

5.3. Experiment Design and Results

The experiment simulates 3D scene-based visualization tasks, including an overview of disaster areas based on low-resolution images, feature recognition based on mid-resolution images and damage assessment based on high-resolution images. In this process, the visualization should ensure that tiles are smoothly refreshed at different locations and pyramid levels when the viewpoint is moving, even under the scenario of multi-client concurrent access. Experiments are designed and discussed in this paper to demonstrate the validity of the on-demand retrieval method when accessing multi-layer image tiles.

5.3.1. Tile Request Efficiency Comparison

An automatic flight route was planned on the surface of a 3D virtual Earth model, as shown in Figure 9. First, the viewpoint flew above China at a height of 5 km; then, it descended to a height of 500 m in the earthquake-stricken area of Ludian County, Yunnan Province. Finally, the viewpoint moved to the area close to the ground in counties in Sichuan Province that were seriously affected by the Lushan earthquake. During flight process, the client must continually request tiles at different locations and resolutions. We counted the differences regarding the number of tile requests and returns and the repeatability of accessed namesake tiles to compare the tile request efficiency with or without tile search optimization.

As shown in Figure 10, as the number of datasets increased, the total amount of general tile requests grew, while the amount and rate of available requests continued to decline. By contrast, the number of optimized tile requests remained steady, and the percentage of available access was maintained at more than 97.8% (see Figures 11 and 12) regardless of the number of datasets that were stored in the database. Due to the lack of automatic identification and filtering of the unavailable tiles, the traditional methods caused the majority of the network to be occupied by the unavailable requests and namesake tiles, which seriously affected the clients' real-time access to the target tiles. In such a scenario, the scene refresh rate decreases significantly as the number of data layers increases. The proposed method effectively solves these problems using semantic annotation and the tile filtering process. Based on the on-demand matching process (see Figure 5), the application server can successfully remove the unavailable layers that are irrelevant to the current viewport so that the number of database queries and invalid tile returns can be greatly reduced. Thus, the optimization improves the bandwidth occupancy of available tiles from 600 KB/s to 20 MB/s to ensure the smooth refreshing of the 3D scene in the clients.

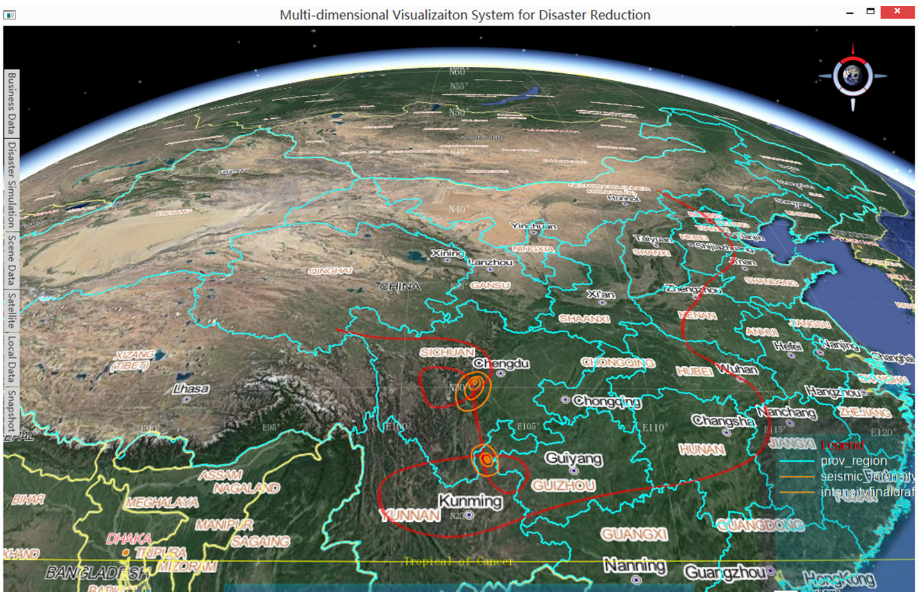


Figure 9. Diagram of the viewpoint flight path.

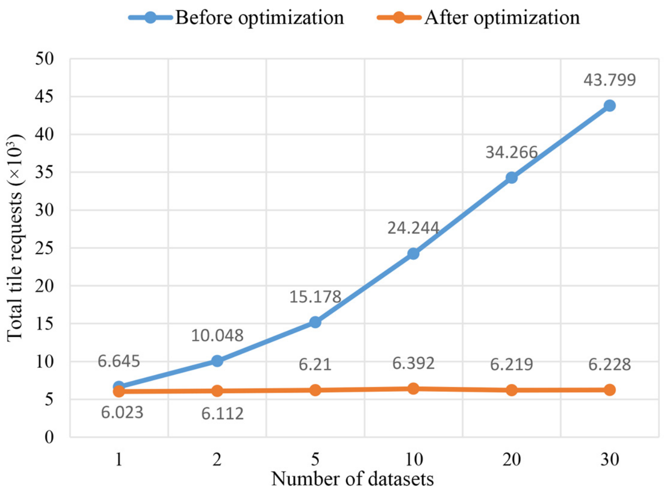


Figure 10. Comparison of total tile requests.

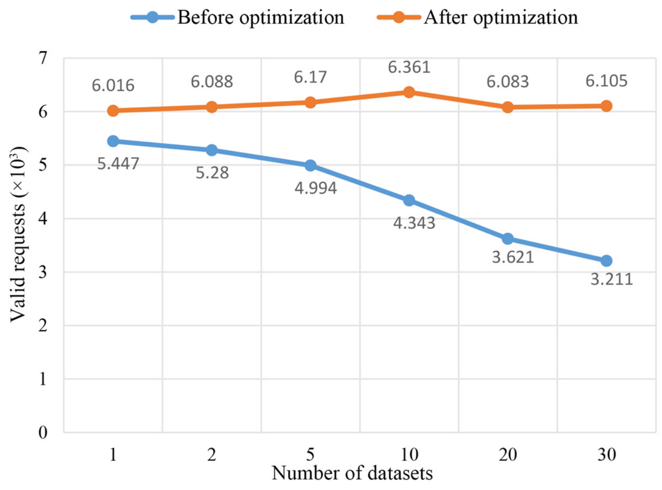


Figure 11. Comparison of valid tile requests.

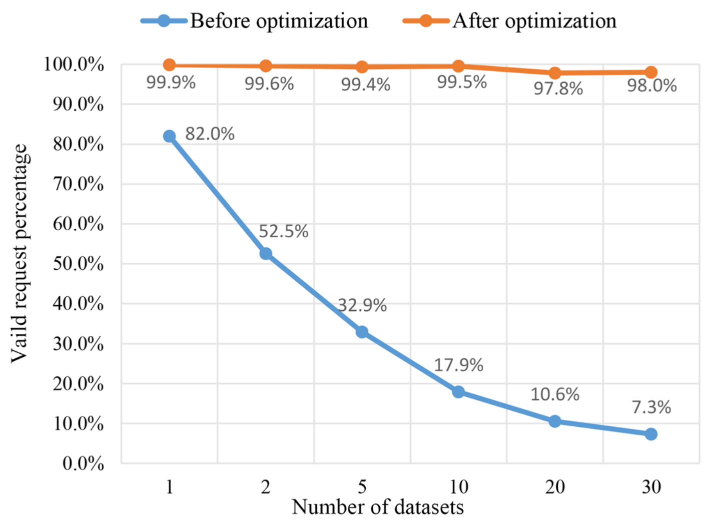


Figure 12. Comparison of valid tile request rates.

5.3.2. Comparison of the Cache Access Efficiency

Twenty clients are established for simultaneous 3D visual browsing of the Ludian earthquake disaster areas, including viewpoint translation, zoom, roaming and flight paths, while concurrently accessing the servers. We tested the server response difference before and after adding the memory cache with the multi-queue LRU replacement method. As Figure 13 shows, the request-response rate of tiles fetched directly from the disk database displays no obvious change, while the response time based on the memory cache design gradually decreases. Moreover, with the increase in the cache capacity, the response speed of the server increases, which indicates that the theme-oriented key-value design makes full use of the retrieval ability of the distributed memory database. This approach can meet the performance requirements of massive image visualization under the condition of multiple clients.

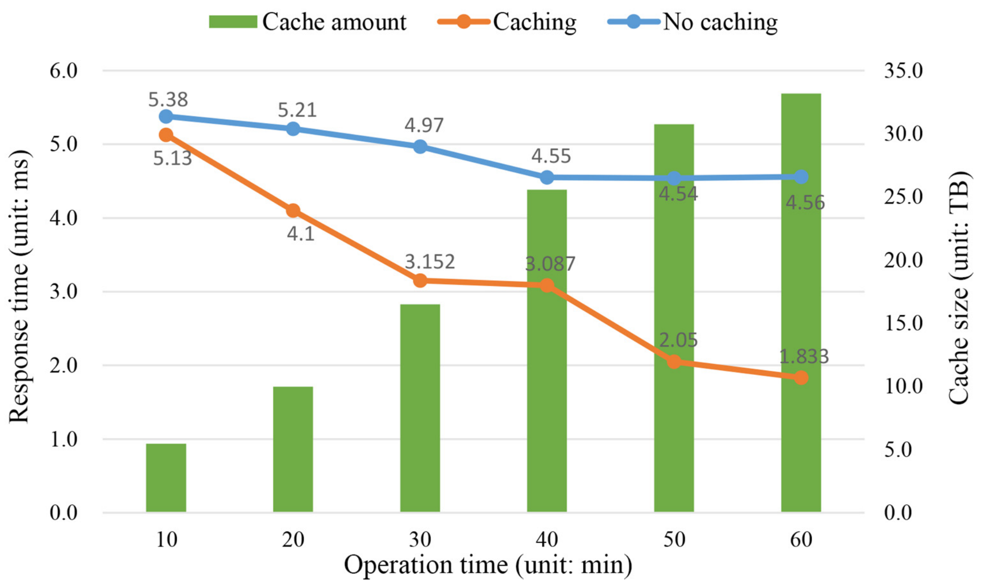


Figure 13. Comparison of the concurrent response efficiency with and without caching.

5.3.3. Multiplexing Performance Comparison

The efficiency of transmission based on HTTP/1.1 and HTTP/2.0 is compared by setting the same viewpoint flight path under the optimized tile retrieval method to ensure that the numbers of available tile requests under different HTTP protocols are the same. We started recording the number of refreshed tiles when the viewpoint flew to the end of the path and the viewport remained. During this period, 235 tiles are requested. The average time consumption of loading tiles based on HTTP/2.0 is 2.73 s, but it takes 10.05 s to load the same tiles using HTTP/1.1. The multiplexing in HTTP/2.0 decreases the time consumption required to establish a connection and support parallel requests in stream form.

Experiments show that the proposed method can improve the efficiency of multi-layer image tile data access in three areas: (1) the on-demand tile matching and filtering process uses semantic annotation and analysis to select the suitable dataset for tile retrieval, significantly narrowing the scope of the database search; (2) the two-layer NoSQL database architecture enhances the response speed of the servers to concurrent access by improving tile retrieval and elimination in the memory cache; and (3) the HTTP/2.0 further improves the transmission efficiency to ensure real-time refresh requirements of client-side visualization tasks. Figure 14 presents the tile loading differences in 2.0 s based on the two methods.

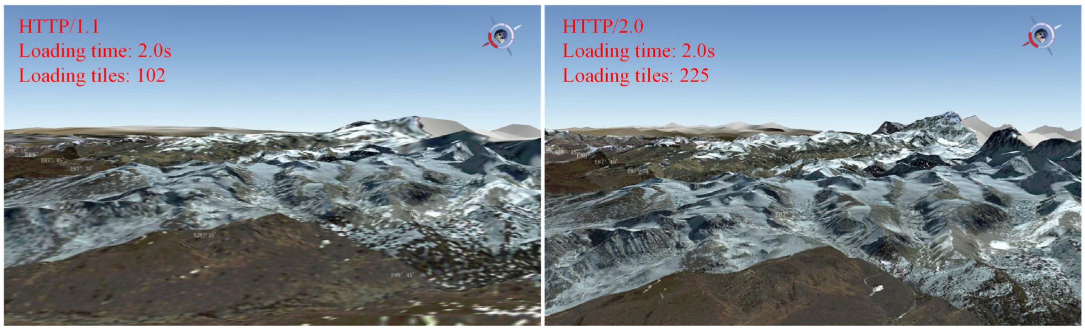


Figure 14. Comparison of tile loading using two different methods.

6. Discussion

The experiment suggests that the on-demand retrieval method can largely improve the efficiency of multi-layer image services in disaster reduction visualization based on the following findings: (1) for multi-dataset tile query, the server can successfully select the most suitable dataset according to the matching process; (2) compared to traditional tile retrieval applications, the effective hit rate of valid tiles is maintained at a high level, regardless of the impact of the number of datasets; and (3) the NoSQL-based data management design improves tile caching and searching on the server side, which supports concurrent access by multiple clients. Since client access to the multi-layer image service is unaffected by the number of datasets, the database could include more datasets for various tasks associated with disaster reduction visualization. Datasets could be labeled with disaster cases and types using theme semantics based on literature knowledge. For example, each disaster could be assigned preferential data sources with an optimal image resolution, and these preferred data sources would be primarily selected during the matching process. Therefore, when the database has stored a sufficient number of datasets for various disasters, the proposed method could be used to analyze the disaster image demands based on the semantic descriptions and automatically select tiles from the most suitable dataset for visualization by the client.

7. Conclusions

This paper proposes an on-demand tile retrieval method based on a hybrid NoSQL architecture to accurately select suitable tiles from multiple layers for diverse visualization in disaster reduction. By building semantic annotations for image datasets, the method can filter invalid layers and find the most available tiles according to the theme, spatial-temporal and resolution aspects of the visualization tasks. The optimization can effectively reduce the number of tile requests from clients and the number of server-side data retrievals to maintain stable transmission of valid tiles from the quantity of stored datasets. Furthermore, the NoSQL-based cluster cache takes advantage of the key-value design for fast tile retrieval, and scheduling optimization further enhances the service ability for concurrent clients. This work has been implemented in the National Disaster Reduction Center of China (NDRCC) and significantly promotes disaster reduction.

Acknowledgments: This work was supported by grants from the National Nature Science Foundation of China (No. 41571390 and 41571392), the National Key Research and Development Program of China (No. 2016YFB0502603) and the National High Resolution Earth Observation System (the civil part) Technology Projects of China (No. 03-Y30B06-9001-13/15). We are grateful to the NDRCC for providing the data, hardware and software environment for this research.

Author Contributions: Linyao Qiu, Qing Zhu and Zhiqiang Du conceived the experiments, designed the method, and wrote the paper; Zhiqiang Du acted as the corresponding author; Yida Fan contributed the experimental data and supplied client and server infrastructures for the experiments; and Linyao Qiu and Meng Wang performed the experiments and analyzed the results.

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

References

1. Serpico, S.B.; Dellepiane, S.; Boni, G.; Moser, G.; Angiati, E.; Rudari, R. Information extraction from remote sensing images for flood monitoring and damage evaluation. *Proc. IEEE* **2012**, *100*, 2946–2970. [CrossRef]
2. Sweta, L.O.; Bijker, W. Methodology for assessing the usability of earth observation-based data for disaster management. *Nat. Hazards* **2012**, *65*, 167–199. [CrossRef]
3. Zhai, X.; Yue, P.; Zhang, M. A sensor web and web service-based approach for active hydrological disaster monitoring. *ISPRS Int. J. Geo-Inf.* **2016**, *5*, 171. [CrossRef]
4. Fekete, A.; Tzavella, K.; Armas, I.; Binner, J.; Garschagen, M.; Giupponi, C.; Mojtahed, V.; Pettita, M.; Schneiderbauer, S.; Serre, D. Critical data source; tool or even infrastructure? Challenges of geographic information systems and remote sensing for disaster risk governance. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 1848–1869. [CrossRef]
5. Grossner, K.E.; Goodchild, M.F.; Clarke, K.C. Defining a digital earth system. *Trans. GIS* **2008**, *12*, 145–160. [CrossRef]
6. Guo, H. China's earth observing satellites for building a digital earth. *Int. J. Digit. Earth* **2012**, *5*, 185–188. [CrossRef]
7. Nativi, S.; Mazzetti, P.; Santoro, M.; Papeschi, F.; Craglia, M.; Ochiai, O. Big data challenges in building the global earth observation system of systems. *Environ. Model. Softw.* **2015**, *68*, 1–26. [CrossRef]
8. Dowman, I.; Reuter, H.I.; Dowman, I.; Reuter, H.I. Global geospatial data from Earth observation: Status and issues. *Int. J. Digit. Earth* **2016**. [CrossRef]
9. Lu, N.; Cheng, C.; Ma, H.; Yang, Y. Global discrete grid systems analysis and comparison. In Proceedings of the 2012 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2012), Munich, Germany, 22–27 July 2012; pp. 2771–2774.
10. Farnaghi, M.; Mansourian, A. Disaster planning using automated composition of semantic OGC web services: A case study in sheltering. *Comput. Environ. Urban Syst.* **2013**, *41*, 204–218. [CrossRef]
11. Miyazaki, H.; Nagai, M.; Shibasaki, R. Reviews of geospatial information technology and collaborative data delivery for disaster risk management. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 1936–1964. [CrossRef]
12. Potere, D. Horizontal positional accuracy of google earth's high-resolution imagery archive. *Sensors* **2008**, *8*, 7973–7981. [CrossRef] [PubMed]
13. Bailey, J.E.; Chen, A. The role of Virtual Globes in Geoscience. *Comput. Geosci.* **2011**, *37*, 1–2. [CrossRef]
14. Ding, Y.; Fan, Y.; Du, Z.; Zhu, Q.; Wang, W.; Liu, S.; Lin, H. An integrated geospatial information service system for disaster management in China. *Int. J. Digit. Earth.* **2014**, *8*, 918–945. [CrossRef]
15. Jongman, B.; Wagemaker, J.; Romero, B.; de Perez, E. Early flood detection for rapid humanitarian response: Harnessing near real-time satellite and Twitter signals. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 2246–2266. [CrossRef]
16. Qiu, L.; Du, Z.; Xie, J.; Qiu, Z.; Xu, W.; Zhang, Y. A real-time visualization method of high resolution remote sensing image bigfiles. *Geomat. Inform. Sci. Wuhan Univ.* **2016**, *41*, 1021–1026.

17. Kiester, A.R.; Sahr, K. Introduction to discrete global grids. *Comput. Environ. Urban Syst.* **2008**, *32*, 173. [CrossRef]
18. Sahr, K.; White, D.; Kimerling, A.J. Geodesic discrete global grid systems. *Cartogr. Geogr. Inf. Sci.* **2003**, *30*, 121–134. [CrossRef]
19. Fekete, G.; Treinish, L.A. Sphere quadtrees: A new data structure to support the visualization of spherically distributed data. In *SC—DL Tentative*; Farrell, E.J., Ed.; SPIE: Bellingham, WA, USA, 1990; pp. 242–253.
20. Xiang, L.; Chen, J.; Gong, J.; Zeng, Z. Fast construction of global pyramids for very large satellite images. *Trans. GIS* **2013**, *17*, 282–297. [CrossRef]
21. Wu, H.; He, Z.; Gong, J. A virtual globe-based 3D visualization and interactive framework for public participation in urban planning processes. *Comput. Environ. Urban Syst.* **2010**, *34*, 291–298. [CrossRef]
22. Yu, L.; Gong, P. Google Earth as a virtual globe tool for Earth science applications at the global scale: Progress and perspectives. *Int. J. Remote Sens.* **2012**, *33*, 3966–3986. [CrossRef]
23. Qiu, L.; Wang, M.; Zhu, Q.; Du, Z. An optimal retrieval method of multi-theme image tiles considering the spatio-temporal semantics. *J. Natl. Univ. Def. Technol.* **2015**, *37*, 15–20.
24. Gui, Z.; Cao, J.; Liu, X.; Cheng, X.; Wu, H. Global-scale resource survey and performance monitoring of public OGC web map services. *ISPRS Int. J. Geo-Inf.* **2016**, *5*, 88. [CrossRef]
25. Li, D.; Zeng, L.; Chen, N.; Shan, J.; Liu, L.; Fan, Y.; Li, W. A framework design for the Chinese National Disaster Reduction System of Systems (CNDRSS). *Int. J. Digit. Earth* **2014**, *7*, 68–87. [CrossRef]
26. Xiao, Z.; Liu, Y. Remote sensing image database based on NOSQL database. In Proceedings of the 2011 19th IEEE International Conference on Geoinformatics, Shanghai, China, 24–26 June 2011; pp. 1–5.
27. Fitzner, D.; Hoffmann, J.; Klien, E. Functional description of geoprocessing services as conjunctive datalog queries. *Geoinformatica* **2011**, *15*, 191–221. [CrossRef]
28. Yue, P.; Zhang, M.; Tan, Z. A geoprocessing workflow system for environmental monitoring and integrated modelling. *Environ. Model. Softw.* **2015**, *69*, 128–140. [CrossRef]
29. Klien, E.; Lutz, M.; Kuhn, W. Ontology-based discovery of geographic information services—An application in disaster management. *Comput. Environ. Urban Syst.* **2006**, *30*, 102–123. [CrossRef]
30. Qiu, L.; Du, Z.; Zhu, Q. A task-oriented disaster information correlation method. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2015**, *1*, 169–176.
31. Sun, S.; Wang, L.; Ranjan, R.; Wu, A. Semantic analysis and retrieval of spatial data based on the uncertain ontology model in digital Earth. *Int. J. Digit. Earth* **2015**, *8*, 3–16. [CrossRef]
32. Yang, P.C.; Wong, D.W.; Yang, R.; Kafatos, M.; Li, Q. Performance-improving techniques in web-based GIS. *Int. J. Geogr. Inf. Sci.* **2005**, *19*, 319–342. [CrossRef]
33. Gudivada, V.N.; Rao, D.; Raghavan, V.V. NoSQL Systems for Big Data Management. In Proceedings of the 2014 IEEE World Congress on Services (SERVICES), Anchorage, AK, USA, 27 June–2 July 2014; pp. 190–197.
34. Han, J.; E, H.; Le, G.; Du, J. Survey on NoSQL Database. In Proceedings of the 2011 6th IEEE International Conference on Pervasive Computing and Applications (ICPCA), Port Elizabeth, South Africa, 26–28 October 2011; pp. 363–366.
35. Kaur, K.; Rani, R. Modeling and Querying Data in NoSQL Databases. In Proceedings of the 2013 IEEE International Conference on Big Data, Silicon Valley, CA, USA, 6–9 October 2013.
36. Naheman, W.; Wei, J. Review of NoSQL Databases and Performance Testing on HBase. In Proceedings of the 2013 IEEE International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC), Shengyang, China, 20–22 December 2013.
37. Grolinger, K.; Capretz, M.A.M.; Mezghani, E.; Exposito, E. Knowledge as a Service Framework for Disaster Data Management. In Proceedings of the 2013 IEEE 22nd International Workshop on Enabling Technologies: Infrastructure For Collaborative Enterprises (WETICE), Hammamet, Tunisia, 17–20 June 2013.
38. Gu, Y.; Wang, X.; Shen, S.; Wang, J.; Kim, J.U. Analysis of Data Storage Mechanism in NoSQL Database MongoDB. In Proceedings of the 2015 IEEE International Conference on Consumer Electronics—Taiwan (ICCE-TW), Taipei City, Taiwan, 6–8 June 2015.
39. Li, R.; Feng, W.; Wu, H.; Huang, Q. A replication strategy for a distributed high-speed caching system based on spatiotemporal access patterns of geospatial data. *Comput. Environ. Urban Syst.* **2017**, *61*, 163–171. [CrossRef]

40. Qin, X.; Zhang, W.; Wang, W.; Wei, J.; Zhong, H.; Huang, T. On-line cache strategy reconfiguration for elastic caching platform: A machine learning approach. In Proceedings of the 2011 IEEE 35th Annual Computer Software and Applications Conference—COMPSAC, Munich, Germany, 18–22 July 2011.
41. Li, R.; Zhang, Y.; Xu, Z.; Wu, H. A load-balancing method for network GISs in a heterogeneous cluster-based system using access density. *Future Gener. Comput. Syst.* **2013**, *29*, 528–535. [CrossRef]
42. Li, R.; Guo, R.; Xu, Z.; Feng, W. A prefetching model based on access popularity for geospatial data in a cluster-based caching system. *Int. J. Geogr. Inf. Sci.* **2012**, *26*, 1831–1844. [CrossRef]
43. Park, D.-J.; Kim, H.-J. Prefetch policies for large objects in a Web-enabled GIS application. *Data Knowl. Eng.* **2001**, *37*, 65–84. [CrossRef]
44. Jaleel, A.; Theobald, K.B.; Steely, S.C.; Emer, J. High performance cache replacement using re-reference interval prediction (RRIP). *ACM SIGARCH Comput. Archit. News* **2010**, *38*, 60. [CrossRef]