International Core Journal of Engineering Volume 7 Issue 3, 2021
ISSN: 2414-1895 DOI: 10.6919/ICJE.202103_7(3).0053

# Remote Sensing Images Metadata Query Based on the Two-Dimensional Graph Intersection Algorithm

Jialiang Shan

Merchant Marine College, Shanghai Maritime University, Shanghai 201306, China.

## Abstract

**"Remote sensing image metadata query" refers to the provision of remote sensing image metadata relevant to an area of interest to the user as accurately and expeditiously as possible. Data Integration Service platform in how to let users according to their own needs, rapid screening and display the corresponding remote sensing metadata is still insufficient. In this study, the solutions to the above problems are studied. In this work, the technologies involved are briefly introduced. After people have a good understanding of how to make map applications, a web-based application approach is proposed, listing the tools and technologies needed. As a result, remote sensing image metadata related to the user's area of interest is successfully represented. The conclusion is that the challenge of quickly filtering out the corresponding remote sensing image metadata information using web browsers has been solved.**

## Keywords

**Remote Sensing Image Metadata; Two-Dimensional Graph Intersection Algorithm.**

## 1. Introduction

The Earth observation satellite has the characteristics of wide coverage, high accuracy of information acquisition, no restriction of airspace boundaries, and no concern for the safety of personnel, etc. It has played an important role in economic development and national defense construction. Many scholars at home and abroad have done a lot of in-depth research on data integration service platform, but how to let users according to their own needs, the rapid screening and display of the corresponding remote sensing metadata is still insufficient. This paper has proposed a solution to the problem, so that users can get the data of the region of interest in the ideal time. In addition, users can enter restrictions in a web browser. The libraries used in this article are completely open source, which greatly reduces the cost of implementing the proposed solution. Web applications are primarily based on JavaScript, and servers are based on Java. The purpose of the Web application is to help users select the corresponding remote sensing image metadata according to the region of interest. This article discusses how to create such a Web application and the challenges of implementing such a solution

## 2. Introducing OpenLayers.js library and intersection algorithm

### 2.1 Introducing OpenLayers.js library

OpenLayers makes it easy to put a dynamic map in any web page. It can display map tiles, vector data and markers loaded from any source. OpenLayers has been developed to further the use of geographic information of all kinds. It is completely free, Open Source JavaScript, released under the 2-clause BSD License. It's more flexible and we can use almost any data source, even Google/Bing and others via API wrappers. There is no dependency on one map provider and no usage restrictions.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202103_7(3).0053

They have strong opensource community, so we can make nice vector editing features, multi-projection support, WMS, WFS and other GIS-friendly APIs.

```
<body>
    <div id="map" class="map"></div>
    <script type="text/javascript">
        var map = new ol.Map({
            target: 'map',
            layers: [
                new ol.layer.Tile({
                    source: new ol.source.OSM()
                })
            ],
            view: new ol.View({
                center: ol.proj.fromLonLat([121.48, 31.22]),
                projection: 'EPSG:3857',
                zoom: 10,
                minZoom: 3
            }),
            loadTilesWhileAnimating: true,
            loadTilesWhileInteracting: true
        });
    </script>
</body>
```

## 2.2 Introducing Two-dimensional graph intersection algorithm

### 2.2.1 Region mapping

There is a two-dimensional space, where the x-axis is longitude and the y-axis is latitude and this space is represented by $H = \{(X, Y) \mid -180° \leqslant X \leqslant 180°, -90° \leqslant Y \leqslant 90°\}$ ssuming that the point p is the boundary point of the observation area whose latitude and longitude coordinates in the geographical coordinate system is $(\lambda_P, \varphi_P)$, and that the point is mapped into the two-dimensional space h, the corresponding points is marked q, the coordinates of q points in space h is $(X_q, Y_q)$, and the corresponding mapping relation is $f : L\,LA \to H, p \mapsto q$ and$X_q = \lambda_P$, $Y_q = \varphi_P$.

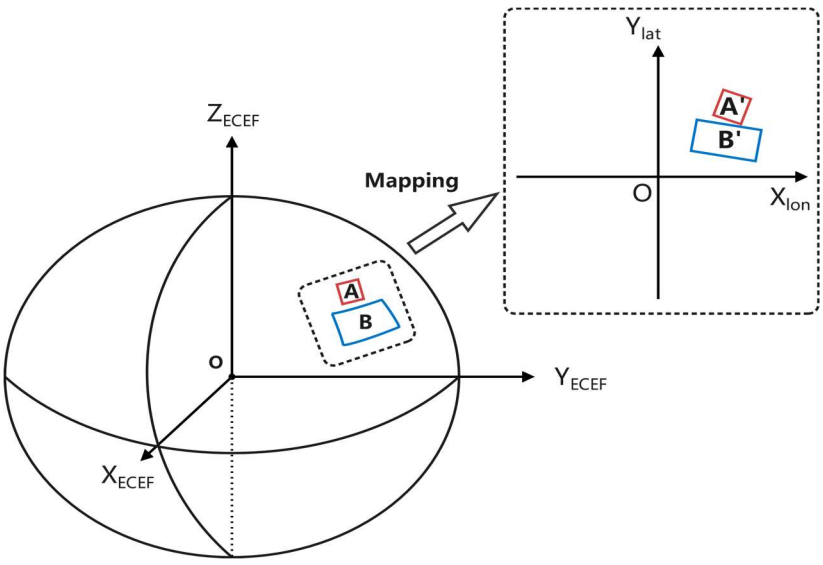


Figure 1. Area mapping schematic

ISSN: 2414-1895 DOI: 10.6919/ICJE.202103_7(3).0053

### 2.2.2 Intersection algorithm

Usually, the region selected by the user is represented as a long rectangle on the sphere, which is the same as the shape displayed on the sphere by remote sensing image. If the region to be observed is non-convex, a number of convex polygons can be obtained by convex decomposition. This paper focuses on the intersection detection of two convex polygons.

(1) If two convex polygons do not intersect, there must be a projection axis on which the projections of the two polygons do not overlap. The main idea is to enumerate all possible axes, and to judge whether the projection of two convex polygons on the projection axis does not intersect, and the normal vectors of each side of the two convex polygons contain all the possibilities of the projection axis, that is, by traversing the edge normal vector of the polygon, we can determine whether there is a gap between two convex polygons on the projection axis. The implementation steps include: According to the coordinates of the vertices of the convex polygon, calculate the edge.

(2) Traversing each vertex of a convex polygon and projecting it onto an axis. The projection of each polygon on an axis can be expressed by the maximum and minimum values of the projection of each vertex on an axis.

(3) If the projections of two convex polygons on the axis overlap, indicating that the two shapes may intersect, the judgment process must be repeated, that is, whether the projections on each axis overlap. If the projections of two convex polygons on a given axis do not overlap, the two convex polygons do not intersect.

As shown in Figure 2. Region a and region b are convex polygons in two-dimensional space after mapping. Region a is the region of interest to users, and Region b is the region covered by remote sensing images. The projection of Region a and b on the axis is expressed as [$a_{min}$, $a_{max}$] and [$b_{min}$, $b_{max}$], respectively, and the order of the two projections on the axis is determined by their minimum values. The relation between two projections satisfies $a_{max} < b_{min}$.So there is a gap between a and b and they do not intersect. As shown in Figure 3, if the projections of regions a and b intersect on an axis, then there is a possibility that a and b will intersect, traversing all the axes, and checking whether the two regions intersect on the axis, that is, the projections on all axes are satisfied $a_{min} \leq b_{max}$ && $b_{min} \leq a_{max}$, then a intersects b.

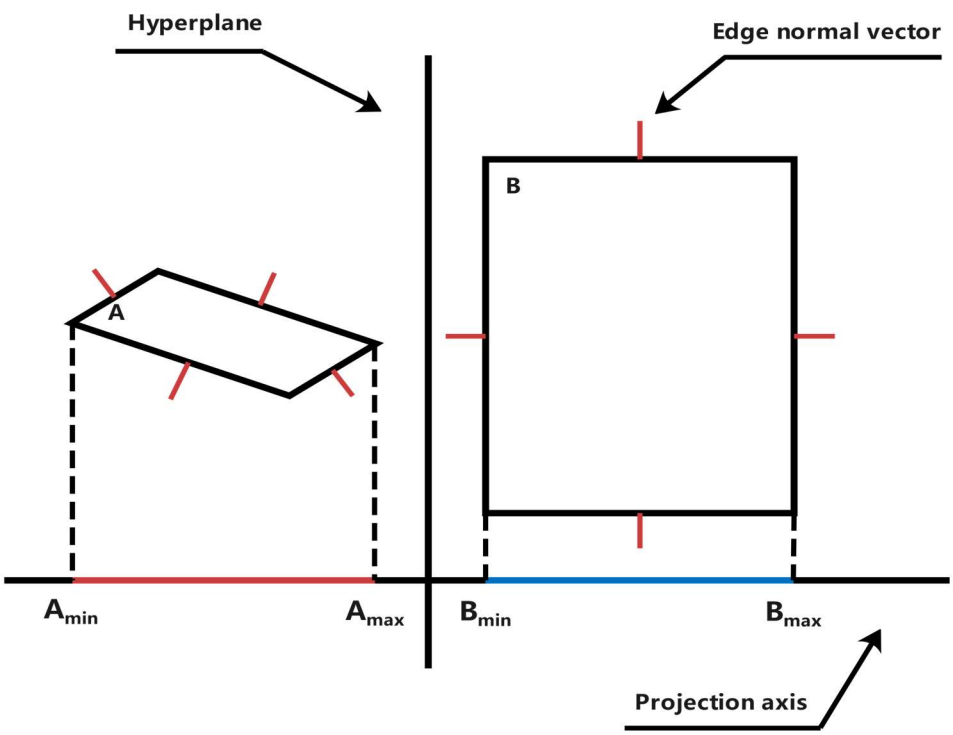


Figure 2. Schematic diagram of intersection algorithm (a)

ISSN: 2414-1895 DOI: 10.6919/ICJE.202103_7(3).0053

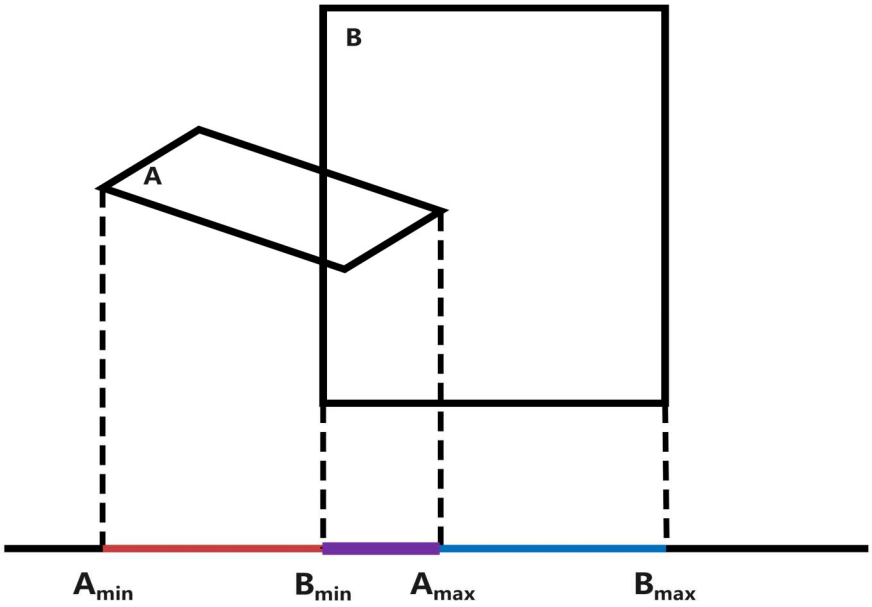


Figure 3. Schematic diagram of intersection algorithm (b)

## 3. Designing the web application

The Web application is loaded through the Tomcat Web server. JavaScript will load the Openlayers and send the data to the server, which uses an intersection algorithm to filter out the metadata of the remote sensing image that intersects the user's area of interest in the database, send the result back to the front-end page and render.

### 3.1 List of software tools used with their function

1) OpenLayers.js for creating 2D map environment.

2) Tomcat acts as a communication between the application and the user.

3) JavaScript handles the front-end operation of the application such as providing.

4) PostgreSQL stores remote sensing image metadata.

### 3.2 Architecture of the web application

The PostgreSQL database pre stores the remote sensing metadata to be tested, and when the user uses the graphics tool provided by OpenLayers to map the area of interest, JavaScript sends the request to the server asynchronously, after receiving and parsing the request, the server reads the latitude and longitude information contained in the image metadata from the database according to the circumscribed polygon vertex data, the image metadata that intersects the given area is filtered, returned in JSON format, and the main field information (such as satellite, sensor, acquisition time, resolution, cloud cover, scene center latitude and longitude coordinates) is front-end. The back-end code is all written in Java, using its powerful Web development capabilities.

### 3.3 Display result

The PostgreSQL database pre stores the remote sensing metadata to be tested, and when the user uses the graphics tool provided by OpenLayers to map the area of interest, JavaScript sends the request to the server asynchronously, after receiving and parsing the request, the server reads the latitude and longitude information contained in the image metadata from the database according to the circumscribed polygon vertex data, the image metadata that intersects the given area is filtered, returned in JSON format, and the main field information (such as satellite, sensor, acquisition time, resolution, cloud cover, scene center latitude and longitude coordinates) is front-end. The back-end code is all written in Java, using its powerful Web development capabilities.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202103_7(3).0053

As shown in Figure 4, the Triangle area in figure is an area of interest to the user. After querying the database containing the test data, two images are filtered to intersect the area. Click on the items in the list to show the user more details, it is convenient for users to select the desired remote sensing image metadata.

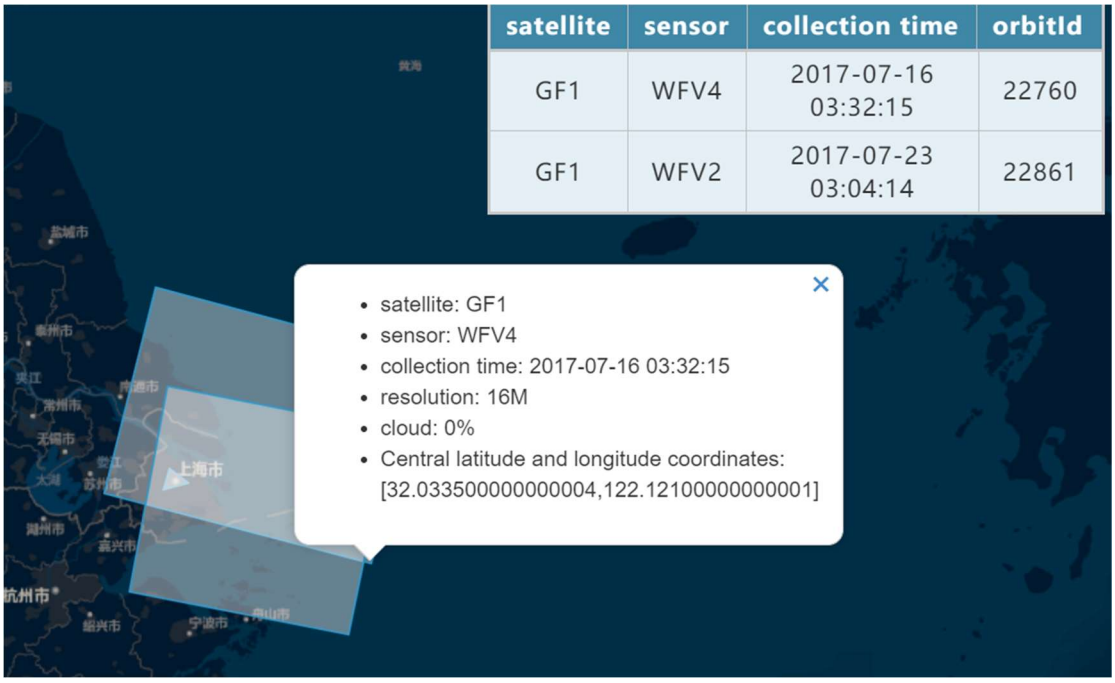


Figure 4. Remote sensing image metadata

## References

[1] Li Yansheng, Ma Jiayi, Zhang Yongjun. Image retrieval from remote sensing big data: A survey. 2021, 67 (prepublish): 94-115.

[2] Binge Cui, Chuanmin Wang, Qing Wang. Remote Sensing Images Data Integration Based on the Agent Service. International Journal of Grid and Distributed Computing:23-30.

[3] Wang Xiaohua, Wang Shudong, Wang Wei. Study on Remote Sensing Image Metadata Management and Issue. IEEE International Geoscience and Remote Sensing Symposium, 2005.

[4] Information on https://www.cheosgrid.org.cn/app/search/search.htm.