applied sciences

MDPI

Article

Camera Geolocation Using Digital Elevation Models in Hilly Area

Zhibin Pan [1], Jin Tang [2], Tardi Tjahjadi [3], Xiaoming Xiao [2] and Zhihu Wu [2,*]

[1] School of Computer Science and Engineering, Central South University, 932 Lushan South Road, Changsha 410083, China; panzhibin@csu.edu.cn
[2] School of Automation, Central South University, 932 Lushan South Road, Changsha 410083, China; tjin@csu.edu.cn (J.T.); xmxiao@csu.edu.cn (X.X.)
[3] School of Engineering, University of Warwick, Gibbet Hill Road, Coventry CV4 7AL, UK; t.tjahjadi@warwick.ac.uk
* Correspondence: wuzhihu@csu.edu.cn

Received: 14 August 2020; Accepted: 21 September 2020; Published: 23 September 2020

Abstract: The geolocation of skyline provides an important application in unmanned vehicles, unmanned aerial vehicles, and other fields. However, the existing methods are not effective in hilly areas. In this paper, we analyze the difficulties to locate in hilly areas and propose a new geolocation method. According to the vegetation in hilly area, two new skyline features, enhanced angle chain code and lapel point, are proposed. In order to deal with the skyline being close to the camera, we also propose a matching method which incorporates skyline distance heatmap and skyline pyramid. The experimental results show that the proposed method is highly effective in hilly area and has a robust performance against noise and rotation effects.

Keywords: enhanced angle chain code; lapel point; distance heatmap; skyline pyramid

1. Introduction

Visual-based geolocalization is an important research area of computer vision. By extracting features from images or videos, and comparing them with the feature databases, the location of the observation can be determined. Different types of visual geolocation methods have been proposed for different environments [1]. For example, in urban areas, image and 3D point cloud are generally used for locating. However, since the obvious characteristics are relatively sparse in the natural environments, it is difficult to directly adopt the geolocation method for urban areas.

In areas with no Global Navigation Satellite System signal, or in outer space with no positioning satellite, locating with skyline is an economic and important method for navigation [2]. For example, by extracting the skyline information from a scene with mountain [3–5] and comparing it with the skyline generated from Digital Elevation Model (DEM) data, the geolocation of the image location can be determined. Locating with skyline has thus become an active research area, which has important applications in unmanned vehicles [6–9], unmanned aerial vehicles [10–15], and other fields.

There are some differences between a photo skyline and a DEM skyline. Any snow cover and vegetation on a mountain interferes with the extraction of skyline features and degrades the contour details of the mountain. Furthermore, some DEM datasets generated from satellite-based data have low resolution and poor precision, which affect the matching of its skyline with a photo skyline. Furthermore, existing methods of locating skyline have been used in an environment where the skyline is far away from mountains and deserts, and the area has few vegetation [16]. However, research has shown that vegetation in hilly areas has a great impact on the feature extraction of the skyline, destroying the original contour details [17,18]. Furthermore, dense hills make the distance between the

skyline and where the photo was taken relatively close (about 1 km), resulting in bad performance of existing geolocalization algorithms.

The contributions of this paper are as follows. First, a new method of generating skyline descriptor named enhanced angle chain code is proposed. Second, a new skyline feature named lapel point is proposed. As far as we know, we are the first to propose such a new feature using the intersection of skyline and ridge line. Third, a new matching method incorporating skyline distance heatmap and skyline pyramid is proposed for locating hilly areas.

The rest of this paper is organized as follows. Section 2 reviews the commonly used skyline description and matching methods. Section 3 introduces the causes of skyline geolocation error in hilly areas. Section 4 presents the proposed new features and new matching method for geolocalization. Section 5 presents experimental results on images of hilly areas captured in the suburb of Changsha, China. Finally, Section 6 presents the conclusions and provides some directions for future research.

2. Related Work

2.1. Representation of Skyline

Although a skyline is a special kind of curve and there are many methods to represent the curve [19,20], the method of skyline characterization is limited. Contour word and Curve Scale Space (CSS) are commonly used to represent a skyline, where contour word is used to locate a skyline [17,21,22] and CSS is used to classify the skyline [23]. Typical methods for locating a skyline use sliding windows to divide the skyline into curvelets [17,21,22]. After sampling the curve data at equal intervals, they are encoded according to the height of the sample points as illustrated in Figure 1. Each point is represented by three bits, and the curvelet is represented as a contour word.

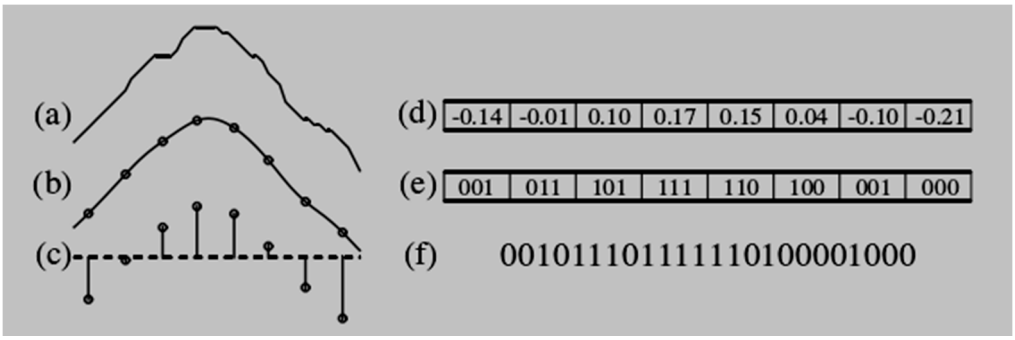


Figure 1. Representing a skyline with contour word. A section of skyline (**a**) is sampled and normalized (**b**–**d**), and each sampling point is encoded with 3 bits (**e**). The contour word (**f**) of the skyline is formed by concatenating the codes of each sampling point.

A curve characterization method based on contour word is simple and effective, which transforms curve matching into binary code matching. Its disadvantage is that it is affected by any rotation of the curve, since a little rotation will result in a significant change to the contour word. Its robustness of this method is poor, especially when the curve is relatively flat and the sampled points has similar height. This is because a little error or noise will have a significant effect on the coding.

The curve characterization method described by the authors of [23] uses the CSS descriptors to represent a skyline. It extracts the skyline and generates the CSS map of the skyline as illustrated in Figure 2. Although this method is robust against rotation of the skyline, the amount of information it provides is less than those provided by other methods. This is because after the length normalization and Gaussian kernel filtering with different widths, only the number and position of the stagnation points of the skyline are recorded. The method only uses the CSS features of skyline to determine whether the skyline is generated by buildings or natural scenes.

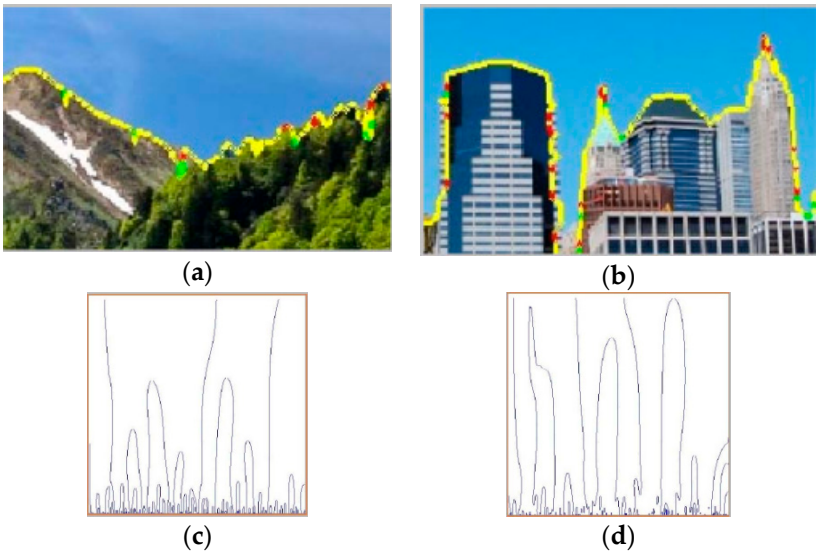


Figure 2. Representing the skylines in (**a**,**b**) by their respective Curve Scale Space (CSS) in (**c**,**d**).

There have been methods [24] for extracting the concavity of skyline as the feature (see Figure 3). One method is as follows. First, the extreme points of curvature in the skyline are found, which are usually in the valley of the skyline. Second, two candidate points are randomly selected on the left and right of each extreme point as the prediction endpoints of the concave curve of the skyline. The positions of the two prediction points are then updated iteratively, until the slopes of the line formed by the two prediction points and the curvature extreme point reached the local maximum value. Finally, the skyline is cut from the positions of each of two endpoints with curvelets, and the two endpoints of each curvelet are normalized to (−1,0) and (1,0) of the coordinates by rotation and scaling. A curvelet is then divided into N equal intervals, and the area between the x-axis and the curvelet is calculated to obtain a n-dimensional feature vector, which is robust to scaling and rotation.

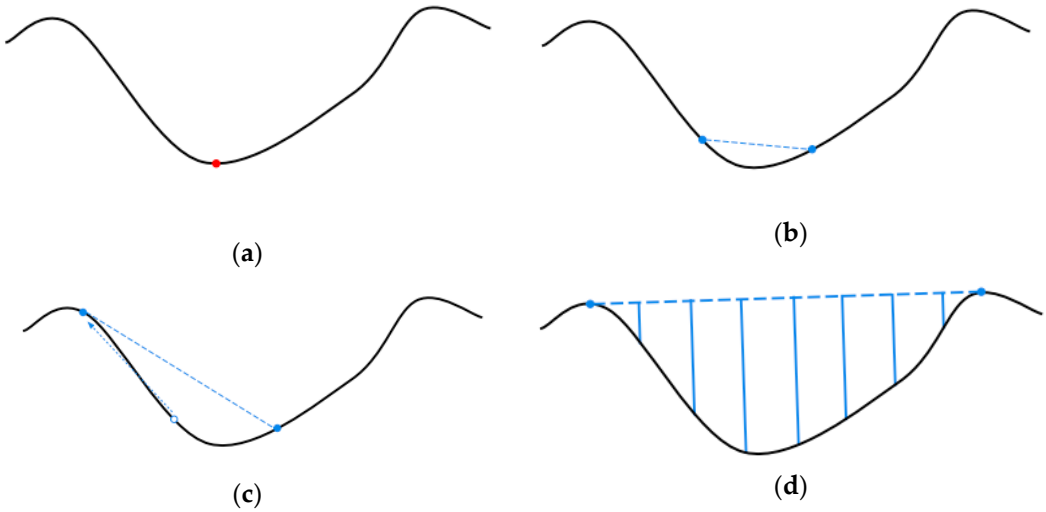


Figure 3. Representing skyline with concavity: (**a**) Find the extreme points of the skyline; (**b**) Set endpoint at initial position; (**c**) Update endpoints iteratively; and (**d**) Coding curves with n-vector.

2.2. The Dataset of DEM

It is important to build the skyline dataset of DEM, i.e., to save the skylines of area to be located. Although some papers [21,24] have proposed that dynamic adjusting the sample density of the DEM

dataset will improve the accuracy of locating, all researchers have used the method of uniform grid sampling to build the DEM dataset, since this method is simpler than dynamic sampling.

The location area of the dataset described by the authors [24] was 10,000 km^2, and the sampling point of the dataset was 1 km. In order to improve the detection speed, the 360° skyline of each sampling point was divided into 24 segments, where each segment had a field of view (FoV) of 30° and the overlapping area between the two adjacent segments was 15°.

The authors of [21] presented experiments in five regions on three continents, each of which used a dataset sampling point of 250 m. The accuracy and efficiency of different sampling density were calculated. The authors of [17] achieved the positioning within 40,000 km^2 in Switzerland. The sampling density of the dataset was 0.001° (111 m) in the north-south direction, and 0.0015° (115 m) in the east-west direction. A total of 3.5 million skylines were produced.

2.3. Skyline Retrieval

Skyline retrieval refers to the extraction of image skyline and finding the most similar skyline in the DEM dataset. Since the retrieval of each skyline in the dataset is very time-consuming, many methods to improve retrieval efficiency are applied to skyline location.

The matching method utilized by the authors of [24] included endpoint matching and shape matching that, respectively, used geometric hash table and k-d tree to improve the matching efficiency. The methods utilized by the authors of [17] and [21] use contour words to represent skyline features. Contour words of all skylines were extracted from DEM dataset, and then sorted by the method term frequency–inverse document frequency (TF–IDF).

3. Difficulties in Locating Skyline Geolocation in Hilly Areas

Compared with mountainous areas and deserts, skyline geolocalization in hilly areas has its own difficulties.

3.1. Detailed Information of Skyline Affected by Dense Vegetation

In hilly area of low altitude, the vegetation on the surface is mainly composed of trees and shrubs. The vegetation has seasonal variations, which cause great interference to the feature of the skyline. Figure 4 illustrates the effects of vegetation on a terrain surface. Although the vegetation does not affect the general trend of the skyline, the vegetation has different effects on the details of the skyline.

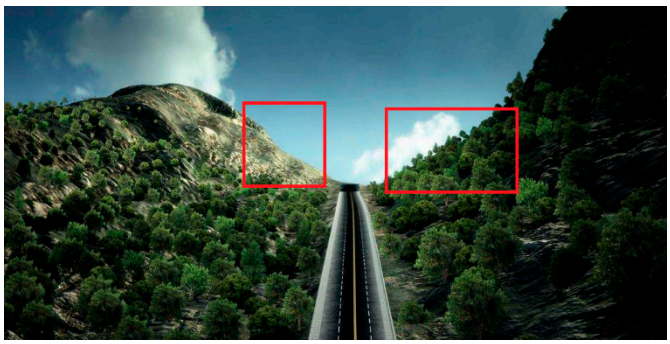


Figure 4. The effects of vegetation on the skyline are illustrated in the areas enclosed by the red rectangles.

3.2. Proximity of Camera to Skyline

For dense hills, the skyline is close to the camera, which leads to locating failure. In skyline geolocation, the nearest sampling point of the DEM dataset is often used as the ground truth of the image taken (as illustrated in Figure 5). Although there is an offset between the ground truth point and image taken position, the skyline features of the ground truth and image are also the most similar. But when the skyline is close to the camera, the difference between the two skylines becomes significant.

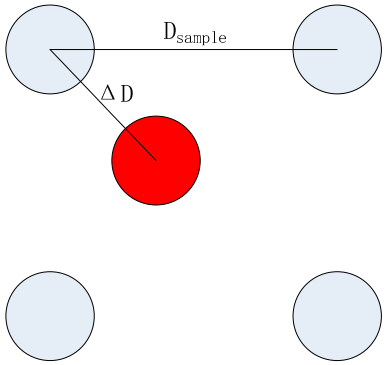


Figure 5. Relationship between the image-taken point and Digital Elevation Model (DEM) sampling points. The red dot denotes the location of image taken point, and the grey dots denote sampling points in the DEM dataset. The sampling point in the upper left corner is the ground truth of the image taken point.

Denoting the distance between two adjacent sampling points as D_{sample}, and the distance between them as ΔD,

$$\Delta D \in \left[0, \frac{\sqrt{2}}{2} D_{sample}\right]. \tag{1}$$

Since ΔD may not be 0, there will be a difference between the image skyline and the ground truth skyline. The difference in skyline caused by this offset is illustrated in the Figure 6.

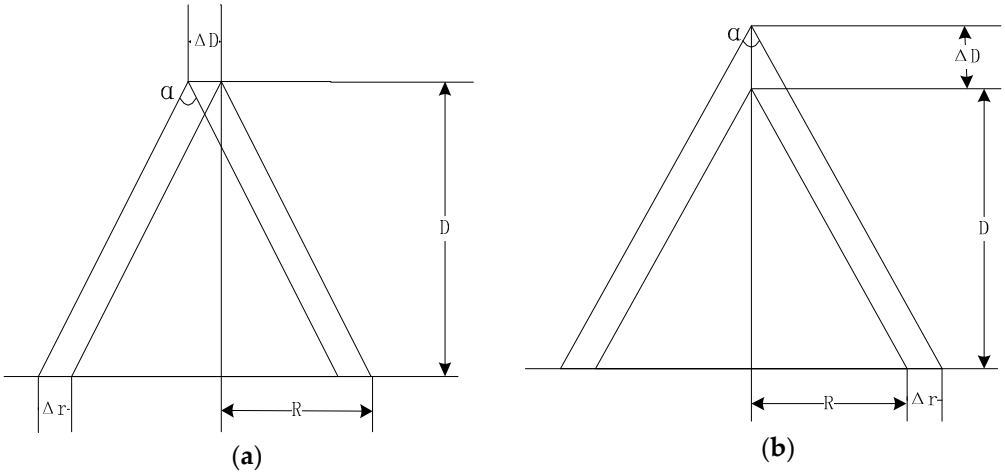


Figure 6. Two differences between image skyline and ground truth skyline: (**a**) The offset direction is orthogonal to the image viewing direction. (**b**) The offset direction is parallel to the image viewing direction. The distance between the image taken point and the skyline is denoted as D. The field of view (FoV) of the camera is α. The error between skylines is ΔD. The length of the skyline is 2R.

When the offset direction is orthogonal to the image viewing direction as in Figure 6a, it is easy to determine that the skyline moves by Δr, and this Δr is equal to ΔD. Since the distance between the camera and the skyline is D, and the FoV of the camera is α, the length of the skyline is

$$2R = 2D \times \tan\left(\frac{\alpha}{2}\right). \tag{2}$$

The percentage of skyline movement due to offset is

$$\frac{\Delta r}{2R} = \frac{1}{2\tan\left(\frac{\alpha}{2}\right)} \cdot \frac{\Delta D}{D}. \tag{3}$$

When the offset direction is parallel to the image viewing direction as in Figure 6b, the change in the length of the skyline is 2Δr, where

$$\Delta r = R \times \left(\frac{\Delta D}{D}\right). \quad (4)$$

When ΔD << D (e.g., in Alps area or desert area), the influence of offset between the image taken point and its ground truth point on the skyline can be ignored. However, when the camera is close to the mountain, such as in a hilly area, the skyline location will be greatly affected.

4. Propose Method

An overview of the proposed geolocalization method for a skyline with dense vegetation in hilly area is shown in Figure 7.

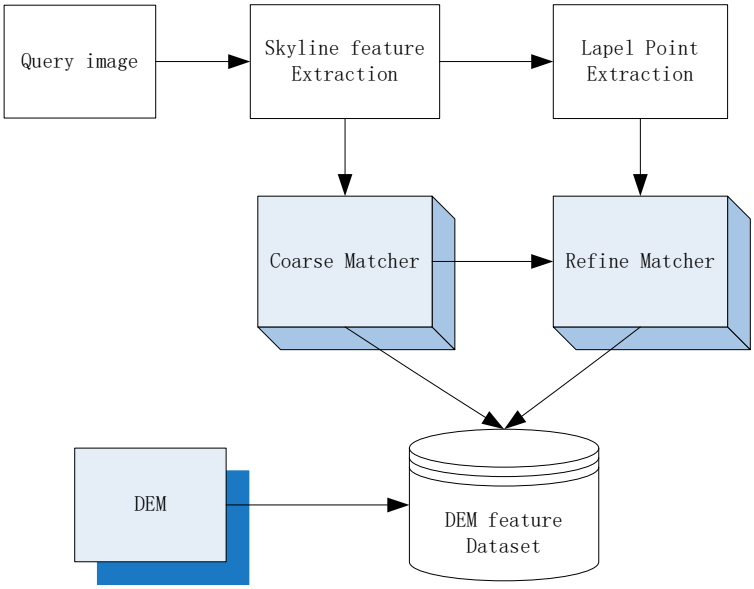


Figure 7. Overview of the proposed method for geolocalization by skyline.

In this method, two new features of skyline are first introduced, which are used for coarse matching and refined matching. Second, a new DEM dataset method is proposed to deal with the situation that the skyline is in hilly area. Finally, the matching method for skylines is introduced, which includes coarse matching and refined matching.

4.1. Enhanced Angle Chain Code

Chain code [25] and BAS (Beam Angle Statistics) [26] are common methods for curve feature, but both methods have certain shortcomings. The use of chain code to represent a skyline is unsatisfactory, because the chain code method is sensitive to noise and is not rotation invariant. When a curve is rotated by a small angle, the encoding of the curve will have a certain change, and any noise on in the curve will also have a large interference to the resulting chain code. The BAS method, on the other hand, samples n points on the closed curve, and calculates the angle between each point and the k-order adjacent points on the left and right sides of a point. Since determining the geolocation of a photo skyline is essentially a matching between a partial curve (an open curve in the photo skyline with width related to FoV, e.g., 24° as illustrated in Figure 8) and an overall curve (360° in DEM), it cannot be applied directly.

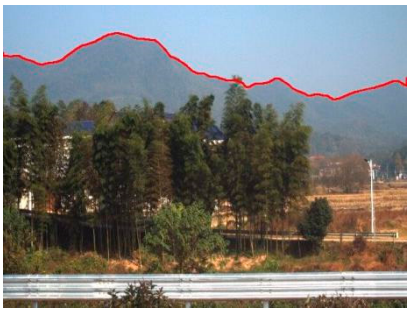


Figure 8. Skyline with FoV of 24°.

In this paper, a new method of representing skyline, named Enhanced Angle Chain code, was proposed based on the combination of chain code and BAS, as follows. First, the 360° DEM skyline is regarded as the standard length, and the skyline of the image is normalized and scaled according to its FoV. Second, set the position of sampled points in the curve (e.g., four sampled points per 1°). Finally, the angle between a sampled point and an adjacent sampled point is calculated. For example, the 1°-order angle is the angle between the current sampled point and the sampled points to its left and right that differ by 1°.

Let the current point be P_{centre}, the reference point on its left be P_{left}, and the reference point on its right be P_{right} (as illustrated in Figure 9). The length of the three sides of a triangle, i.e., a, b, and c, are given by the Euclidean distance. According to the cosine theorem, the angle θ between P_{centre} and the left and right reference points is

$$\theta = \arccos\left(\frac{c^2 - a^2 - b^2}{2ab}\right). \tag{5}$$

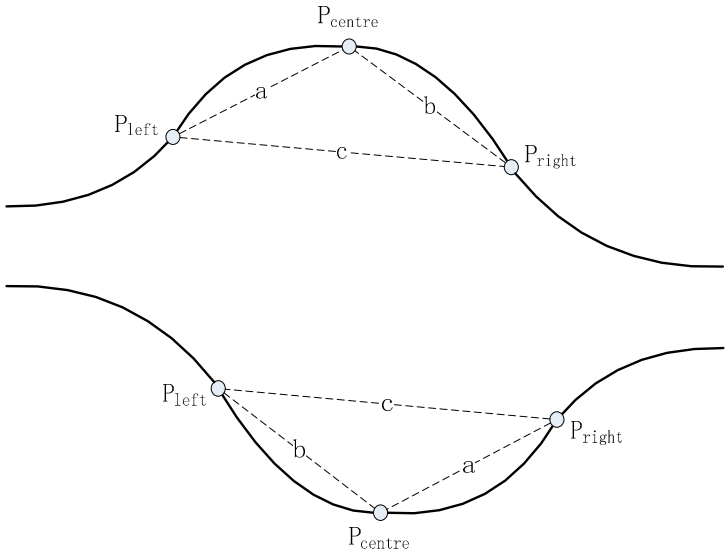


Figure 9. Peaks and valleys have the same calculation results. P_{centre} is the selected point, V_{left} and V_{right} are adjacent points of the selected point, and a, b, and c are lengths of three sides of a triangle. Note that the angles of the triangle are related to the length of three sides.

Considering the triangle parameters corresponding to mountain and valley are the same, the θ value can be calculated and limited according to Table 1.

Table 1. Calculation method of mountain and valley.

Terrain	θ Value	Judgments Based [1]
Peak	$\arccos\left(\frac{c^2-a^2-b^2}{2ab}\right)$	data $\frac{P_{left}.y+P_{right}.y}{2} \geq P_{center}.y$
Valley	$2\pi - \arccos\left(\frac{c^2-a^2-b^2}{2ab}\right)$	$\frac{P_{left}.y+P_{right}.y}{2} < P_{center}.y$

[1] $P.y$ means the y value of the point P_*.

In order to facilitate the comparison, we used the 32-based chain code to encode the feature angles of skylines and transformed the curve matching into a string matching. A 32-neighbour representation was proposed by dividing the 360° into 32 subintervals (as illustrated in Figure 10) that are represented by "0~9" and "A~V", and each subinterval corresponds to 11.25°. The enhanced angle chain code is a more accurate descriptor.

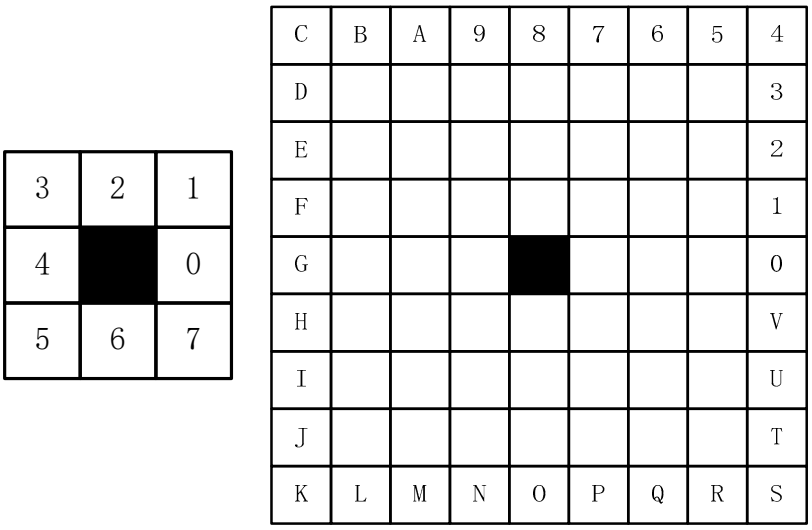


Figure 10. The directions of chain code (**left**) and enhanced angle chain code (**right**).

Since the 32-base encoding was adopted, the encoding corresponding to the θ angle at P_{centre} is given by

$$\text{Code}_\theta = \left\lfloor \frac{\theta}{\frac{\pi}{16}} \right\rfloor. \tag{6}$$

In order to reduce the likelihood of coding different tilted versions of the same skyline, the normal vector of the curve was recorded at each sampled point of skyline (as illustrated in Figure 11), and 32-bit chain code was used for coding. Given the slope of $P_{left} \cdot \overrightarrow{P}_{right}$,

$$\text{Slope}_{\text{chord}} = \text{atan}\left(\frac{y_{right} - y_{left}}{x_{right} - x_{left}}\right). \tag{7}$$

and

$$\text{Slope}_{\text{curve}} = \begin{cases} \text{Slope}_{\text{chord}} + 90° & when\ Slope_{chord} \leq 90° \\ \text{Slope}_{\text{chord}} - 90° & when\ Slope_{chord} > 90° \end{cases}. \tag{8}$$

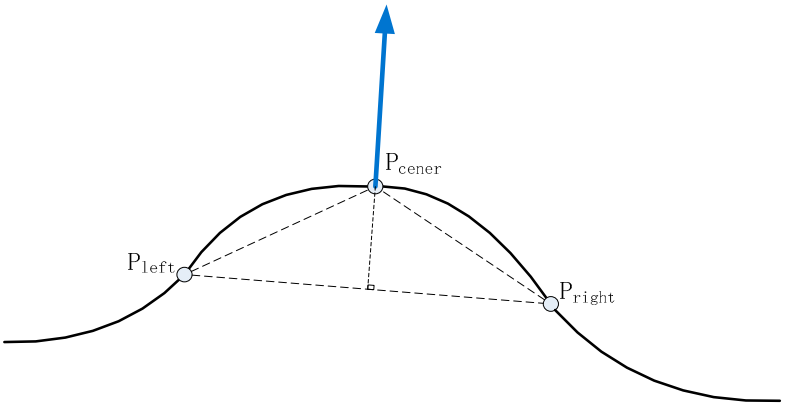


Figure 11. Computing the normal vector of a curve. P_{centre} is the selected point, and V_{left} and V_{right} are adjacent points of the selected point. The blue arrow is the normal vector of the curve at P_{centre}.

Therefore, after extracting the skyline using supervised learning [27] and saving it as a list (as illustrated in Figure 12), a skyline comprises two Enhanced Angle Chain codes named curvature Enhanced Angle Chain code and normal Enhanced Angle Chain code to describe its features (see Table 2). Curvature enhanced chain code records the angle between the sampled point of the curve and the surrounding sampled points. Normal Enhanced Angle Chain code records the normal vector angle at each sampled point of the curve.

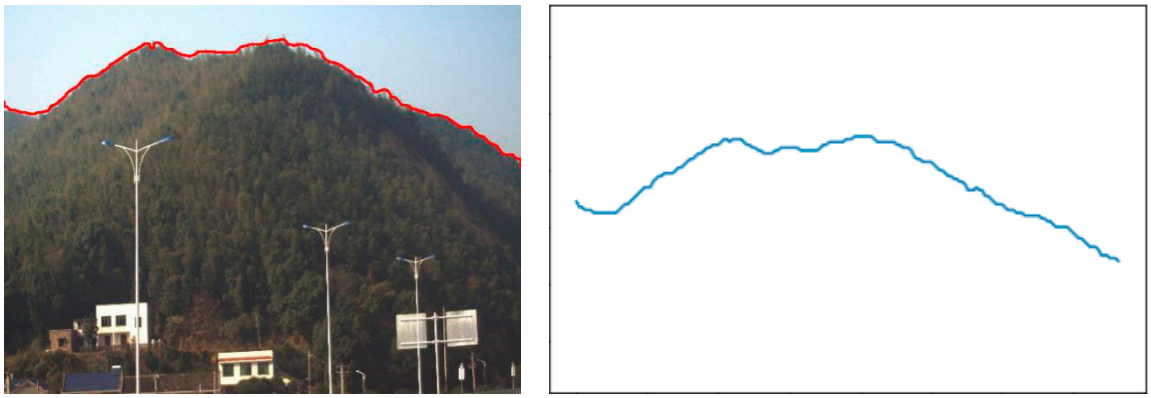


Figure 12. Storage method of skyline. Use a list (**right**) to record the skyline data and the length of the list is the width of the photo (**left**). List the index of each element according to its corresponding position on the image, and the position of each element is the height of the point on the image.

Table 2. The Enhanced Angle Chain Code (2°-order) of the skyline in Figure 12.

Subfeature Name	Subfeature Content
curvature enhanced angle chain code	D-E-F-G-G-G-G-G-H-I-J-K-J-J-H-G-E-E-E-E-F-E-E-F-G-G-H-I-I-I-I-I-H-H-G-G-G-G-G-G-F-G-F-E-F-F-F-F-G-H-H-H-H
normal enhanced angle chain code	9-A-A-A-B-A-A-A-A-9-9-8-8-8-8-7-7-7-7-8-8-8-8-8-8-8-8-7-7-6-6-6-5-5-5-5-5-5-5-5-5-5-5-5-5-5-6-5-5-5-5-5-5

4.2. Lapel Point

Lapel refers to the direction in which the two collars overlap. In different periods of ancient China, due to the different requirements of different dynasties, there were different requirements for left lapel and right lapel (see Figure 13).

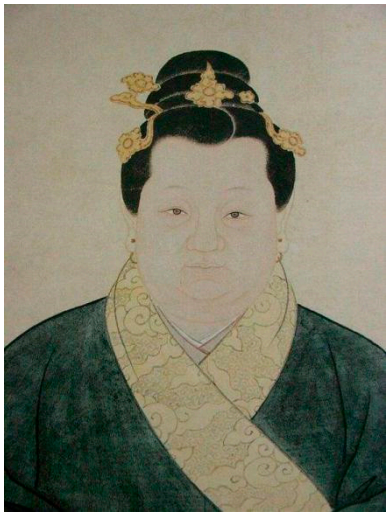


Figure 13. Left lapel dress in ancient China.

Right lapel means the left collar is covering the right collar, where its front looks like a small letter y at the edge of the collar. The direction of left lapel is opposite to the right lapel, and the collar opening is left (as illustrated in Figure 14).

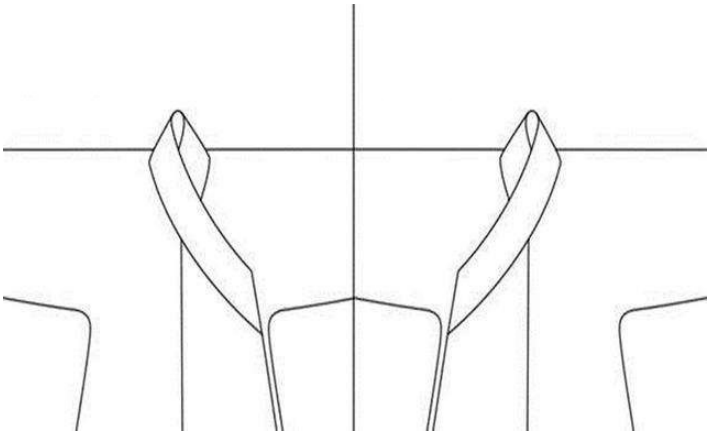
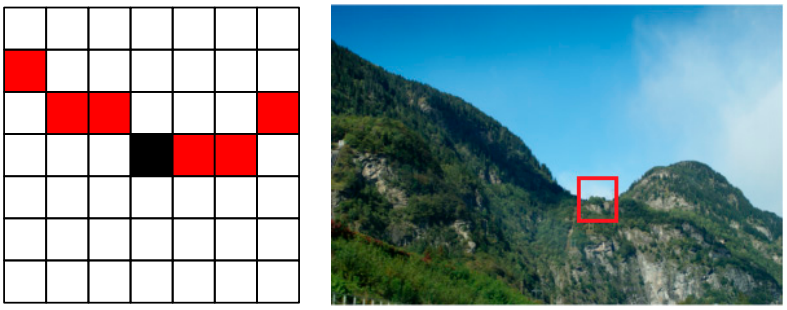


Figure 14. Left lapel and right lapel.

The lapel point of skyline refers to the intersection of ridge line and skyline and belong to both skyline and ridge line (as illustrated in Figure 15).



(**a**)

Figure 15. *Cont.*

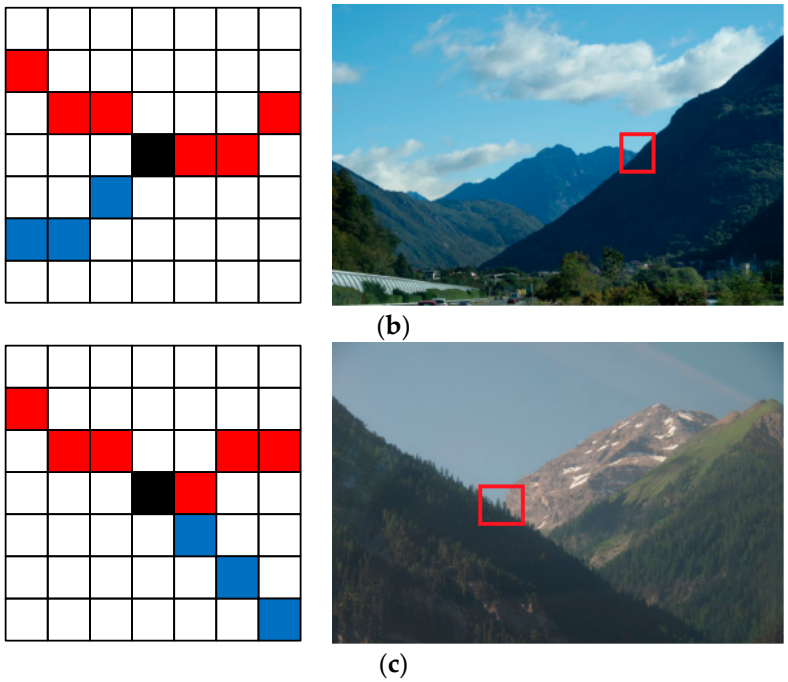


Figure 15. The lapel point of skyline: Row (**a**) Normal point of skyline; Row (**b**) Left lapel point of skyline; and Row (**c**) Right lapel point of skyline. The black dots are the skyline points to be detected. The red dots are the skyline points and the blue points are the ridge line points.

4.2.1. Extracting Lapels from Images

Since the image and DEM are of different data types, the methods of extracting the lapel point are also different, as follows. First, skyline and ridge line are extracted from the image using methods of semantic segmentation, such as U-Net. Second, the sliding window is used to traverse each pixel on the skyline and extract its immediate eight neighbor. Then, each ridge point in the eight neighbors are marked as candidate points, and the trend of the ridge line to which the candidate point of the ridge line belongs is found through the dynamic programming algorithm. Finally, by comparing the x-axis coordinates of the candidate points with the midpoint of the ridge line, the type of the lapel points is determined, as in Table 3.

Table 3. The type of lapel points in a picture skyline.

Characteristics	Type
$X_{mean} > X_{candidate}$	Left lapel point
$X_{mean} \leq X_{candidate}$	Left lapel point

4.2.2. Extracting Lapels from DEM

Unlike image data, it is easy to use DEM to obtain the depth information of skyline while extracting the skyline. The jumping point of DEM skyline refers to two adjacent points, and the distances between the two skyline points and the observation point are quite different. The reason is similar to that of the image skyline lapel point. The distances between the skyline of the same mountain range and the observation point are relatively small, while the distances between different mountains and the observation point are quite different due to the different spatial positions.

The depth information of the DEM skyline can be obtained from the coordinate points corresponding to the observation points and the skyline points (see Figure 16). The depth jump points

of the DEM skyline were obtained by calculating the first-order derivative of the depth value of the skyline, i.e.,

$$\text{Derivative}_i = \begin{cases} \text{Depth}_i - \text{Depth}_{i-1} & if\ abs(Depth_i - Depth_{i-1}) > threshold \\ 0 & if\ abs(Depth_i - Depth_{i-1}) < threshold \end{cases}. \quad (9)$$

where the threshold value corresponds to points with great change in the depth of the skyline (see Figure 17). The left and right sides of these jumping points belong to different mountains, corresponding to the lapel points in the same position in the image skyline.

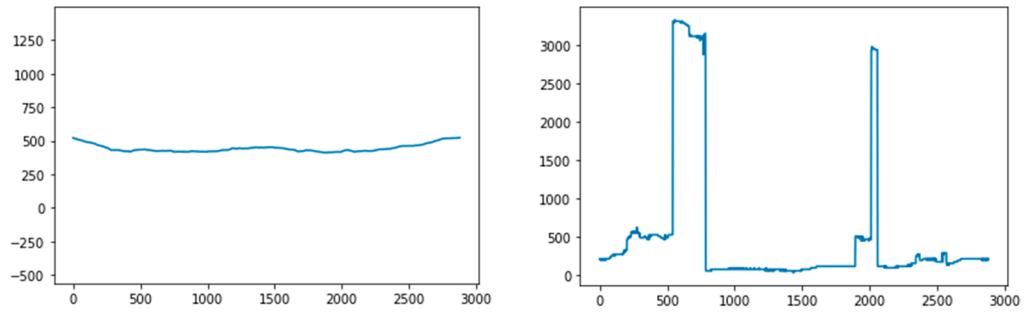


Figure 16. DEM skyline (**left**) and its depth (**right**).

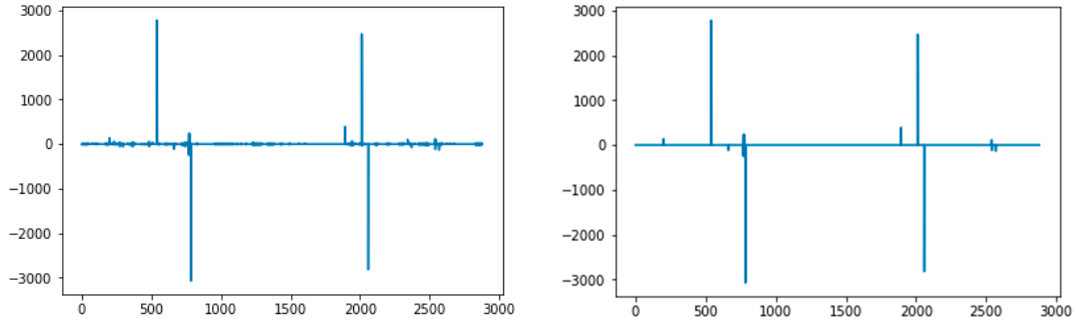


Figure 17. Example of first-order derivative of skyline depth (**left**) and the effect after thresholding (**right**).

There are differences between DEM and image. When the distance between two lapels of DEM is less than the width threshold, it indicates that only a small part of a mountain range appears in the skyline, and thus may not be recognized. When the two lapels are of the same type, they will be merged into one lapel point. When the two lapels are of different types, the two lapels will be offset. The method of saving jump points in DEM is

$$[\text{Index, Type}], \quad (10)$$

where Index is the x-axis coordinate of the lapel point in the DEM rendering image, and Type is the type of the lapel point.

4.3. Building DEM Skyline Feature Database

The DEM skyline database was used to record DEM skyline features. Through the efficient management of features, the efficiency of image retrieval in the database can be greatly improved. For close skyline in hilly area, the heatmap of skyline distance was added to improve the locating accuracy. An overview of the database building method is shown in Figure 18.

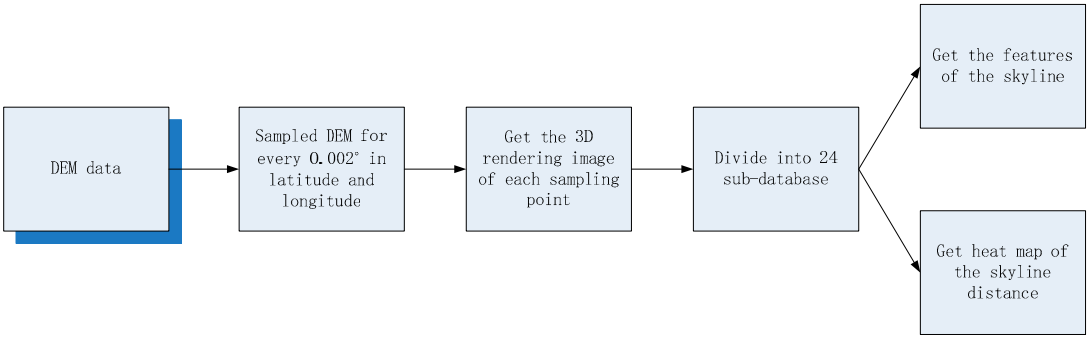
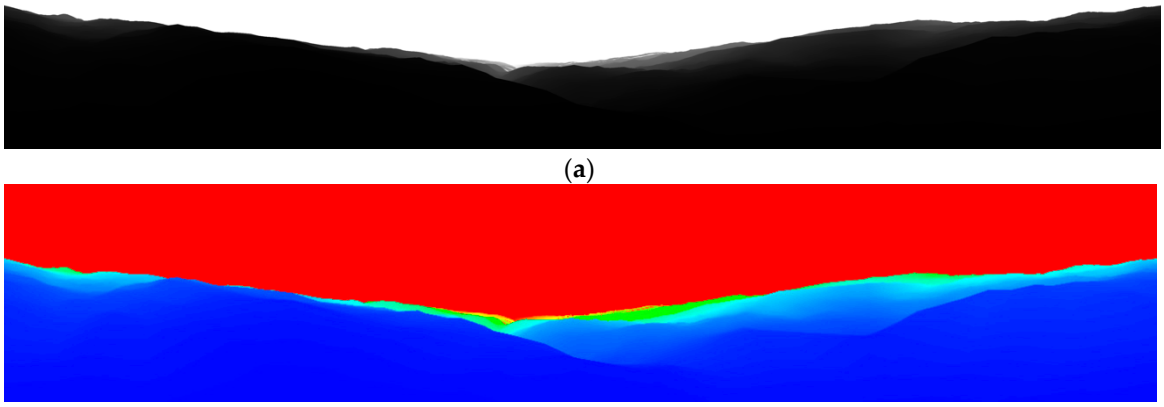


Figure 18. Flowchart for constructing the DEM database.

4.3.1. Sampling and Generating 3D Rendering Image

The DEM data sampling density used in this dataset was 10 m, and we obtained the DEM skylines for every 0.002° in latitude and longitude and from 1.80 m above the ground, giving a total of 71 × 71 DEM skylines. Figure 19 illustrates the rendering of a DEM sampled point.



(**a**)

(**b**)

Figure 19. 360° rendering generated from a DEM sampled point. (**a**) Original greyscale image. The leftmost end of the image is in the north direction, which is unfolded in a clockwise direction. The middle of the picture is due south, and the far right is due north. The white part of the image is the sky, and the lower part shows the distance between the mountain and the observation point displayed with different grey levels. The closer the mountain is, the higher the grey value. The lower boundary of the white area is the skyline, and where the grey level of the lower half changes is the ridge line. (**b**) The pseudo color image was generated from the original greyscale image, which is convenient for manual observation and judgment.

4.3.2. Divide into 24 Subsections

Similar to the method described by the authors of [25], at each sample point, we rendered twenty-four 30° images at 15° offsets, covering the full 360° panorama at the location. Thus, the DEM skyline database was divided into 24 subdatabases according to angles, thereby accelerating the speed of retrieving and matching of the corresponding skyline.

4.3.3. Get the Features of the Skylines

The skyline features were extracted from each subdatabase, including the enhanced angle chain codes and skyline lapel points.

4.3.4. Get Heatmap of the Skyline Distance

Unlike other methods, the heatmap of DEM skyline distance was the first to be proposed for hilly area. For each subdatabase, the minimum distance of each sampling point skyline was extracted, and the distance heatmap was generated (as illustrated in Appendix A Figure A1).

As mentioned in Section 3.2, as the distance between the skyline and the camera becomes closer, the error between the location where the photo was taken and the ground truth will have an increasing impact on locating. Therefore, the distance in the heatmap was classified (as illustrated in Table 4 and Figure 20), and different locating strategies were adopted for different categories.

Table 4. Classification conditions of heatmap.

Type Code	Distance Range
00	[0, 800]
01	(800, 1500]
10	(1500, +∞)

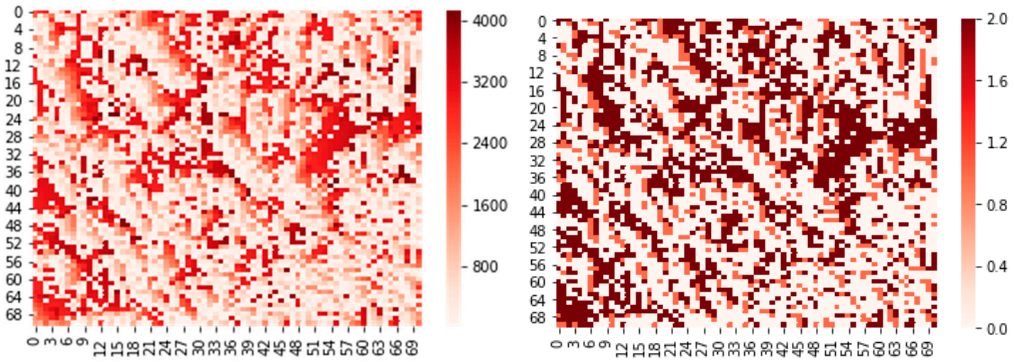


Figure 20. Heatmap (**left**) and distance classification map (**right**).

4.3.5. Coarse Matching

In the proposed coarse matching, skyline features are extracted from the image, and the corresponding DEM subdataset is selected according to the heading angle (obtained by compass, etc.). For hilly areas and the heatmap of skyline, the image skyline pyramid was proposed to improve the geolocalization accuracy of the skyline. According to the distance between adjacent skyline in DEM database and the distance between camera and skyline, the skyline of the original image was enlarged and reduced by 10% and 20%, respectively (as illustrated in Figure 21).

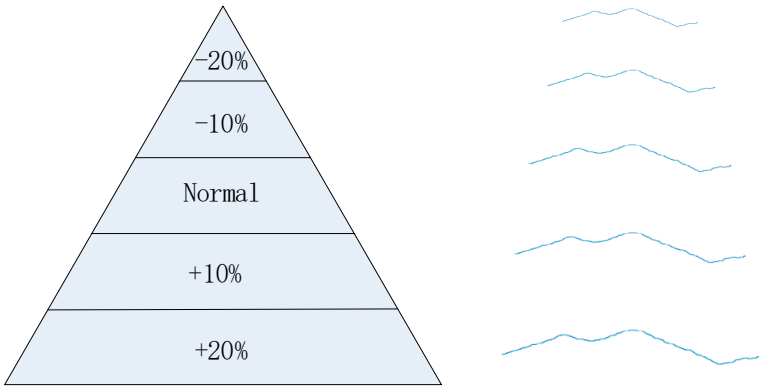


Figure 21. Image skyline pyramid (**left**) and the corresponding skylines (**right**).

When extracting the Enhanced Angle chain code, the original camera FoV is used as the skyline of the original image, and the zoomed FoV is used for the skyline of other sizes.

The common edit distance is used to determine the similarity of two skylines. The magnitude of edit difference and the mean of edit distance are used to measure the difference between two skylines. Unlike the traditional edit distance, the difference in distance of skyline matching is not only related to the number of different characters, but also related to the similarity of different characters. If the two strings to be compared are *str1* and *str2*, and a_i and b_i are, respectively, the ith character on *str1* and *str2*, then the average distance between the two strings is

$$Dis(str_1, str_2) = \frac{\sum Diff(a_i, b_i)}{length}, \tag{11}$$

where

$$Diff(a_i, b_i) = \frac{0.5 \times abs(Curv\_a_i - Curv\_b_i) + 0.5 \times abs(Normal\_a_i - Normal\_b_i)}{32} \tag{12}$$

computes the degree of difference between two characters, and the range of value is [0, 1). A value of 0 means the two characters are exactly the same, and a value of 1 means the difference between the two characters is very large.

The image skyline features are compared with the features of each point in the subdataset, and the minimum error between the image skyline and each sampling point in the subdataset is recorded. For different types of heatmaps, different comparison strategies are adopted.

When the skyline is far away from the sampling point, the features of the original image skyline and the skyline features of the sampling point are used for comparison, and the error is recorded. When the distance between the skyline and the sampling point is moderate, the skyline of the original image and the skyline compressed and expanded by 10% are compared with the skyline features of the sampling points respectively. The minimum of the three error values is recorded. When the skyline is close to the sample point, the skyline of the original image and the skyline compressed and expanded by 10% and 20%, are compared with the feature of the sampling point. The minimum value of the five error values is recorded. The process of coarse location is shown in the Figure 22.

4.3.6. Refined Matching

Coarse matching can obtain a location result which usually has about 200 candidates (as illustrated in Figure 23). Refined matching was used to improve the coarse matching results and eliminate any unreasonable candidate points (as illustrated in Figure 24). Coarse matching only found similar points in the DEM skyline database through skyline contour features, and refined matching eliminates interference results through location, type, and sequence of lapel points.

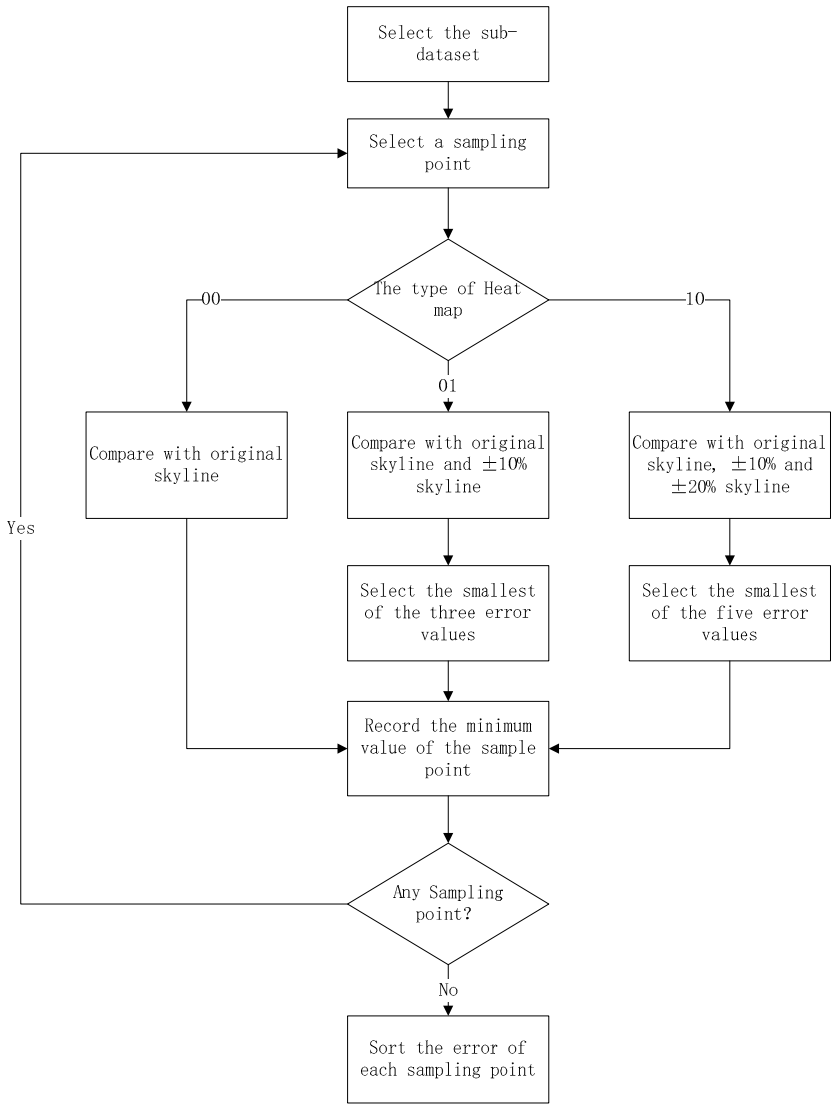
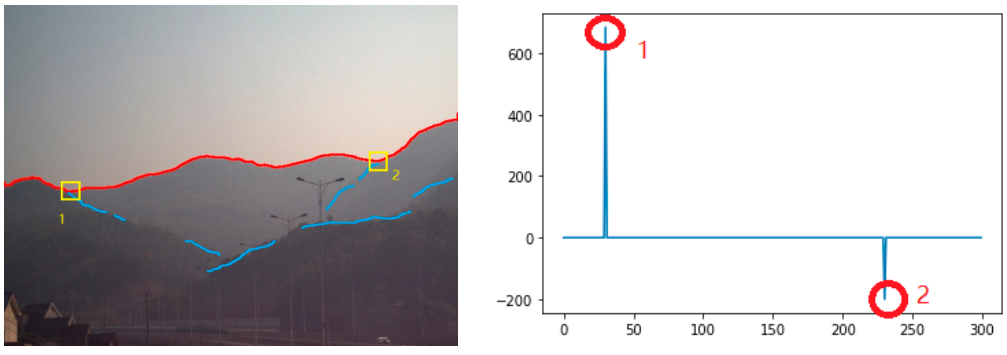
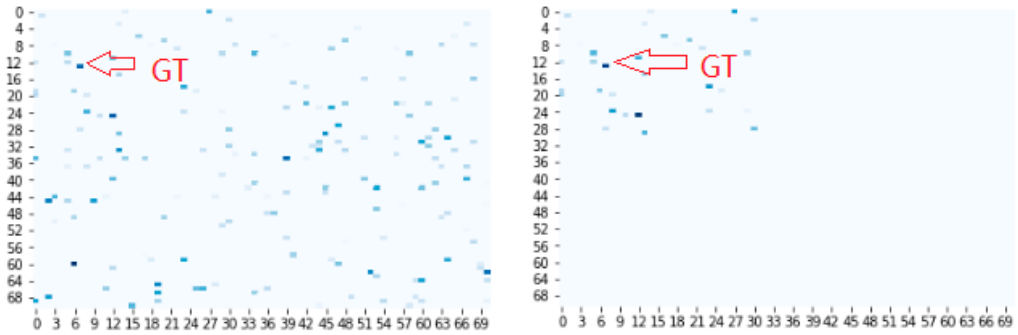


Figure 22. Flowchart of coarse matching.



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

Figure 23. The candidate locations are verified by the feature of lapel points. The retrieved image contains two lapels points (**a**), and the DEM candidate point also contain two similar lapel points at the position where the skyline coincides (**b**). This improves the reliability of the location where the image was taken.



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

Figure 24. Coarse matching (**a**) and refined matching (**b**) The blue dots indicate the candidate points for localization, and the dark color means that the coarse matching error was small. lapel points can greatly improve the localization accuracy.

5. Experiment and Analysis

In order to demonstrate the effectiveness of the proposed method, we first performed experiments on the effects of noise, pitch, rotation, and other factors on the proposed method. We then compared the performance between the proposed method and the art-of-state methods on our dataset.

5.1. Dataset

The dataset was collected in a hilly area (as shown in Figure 25), which was about 200 km^2 (28.08° N, 112.69° E~28.22° N, 112.83° E), located in the suburb of Changsha City, China. In this region of interest (ROI), the skylines change greatly, and the vegetation is dense. However, the manmade buildings are few and have little interference on the trend of the skylines.

Figure 25. The region of interest (ROI) considered is denoted by the semitransparent polygon.

The DEM dataset with an accuracy of 10 m was used to establish the DEM skyline database. Meanwhile, more than 200 photos were captured with a camera (focal length was 8 mm, FoV was 24°, and the resolution was 1920 × 1080), which was fixed on the platform about 180 cm above the ground. Figure 26 shows an example photo and Table 5 shows the longitude and latitude, attitude angle, and other information during the photoshoot.

Figure 26. A photo taken in the experimental area.

Table 5. Location information of the image label.

Parameters	Values
pitch	4.49°
row	−0.26°
yaw	356.42°
latitude	28.15135°
longitude	112.7574°
satellites	10

5.2. Localization Performance

The experiment involved 284 photo skylines and 5041 DEM skylines in an area of about 200 km^2. The values of Top_1 to Top_100 items in the candidate list of correct locations were calculated. The photo skylines and the ground truth of DEM skylines were different in some details due to vegetation effects and error between locations (as illustrated in Figure 27).

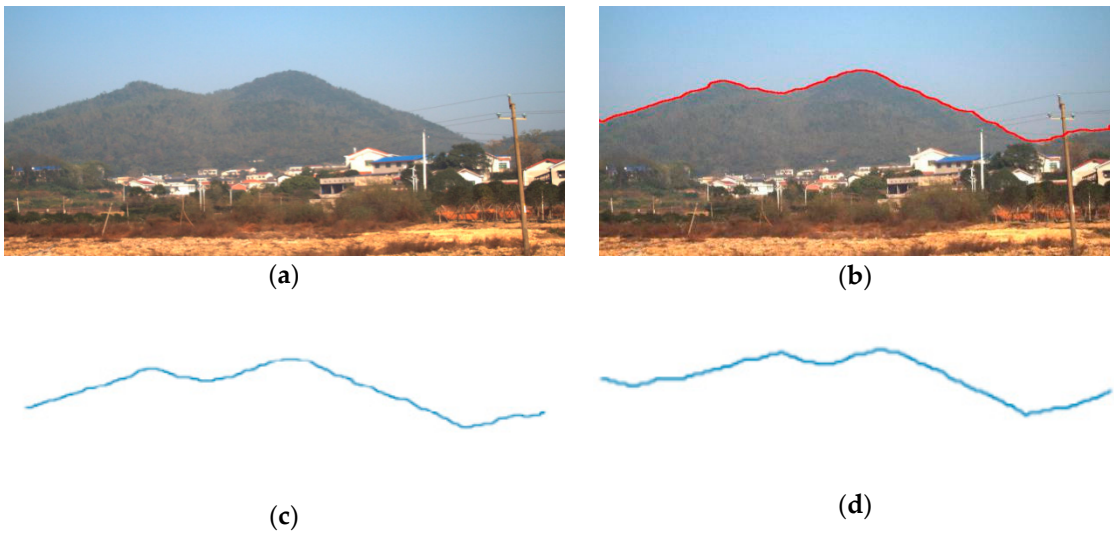


Figure 27. Some differences between the photo and DEM skylines: (**a**) Photo skyline; (**b**) Extracted skyline (denoted in red) superimposed on the photo skyline; (**c**) Extracted skyline; and (**d**) The ground truth of DEM skyline.

The skyline features of each image were extracted in turn, and then retrieved in the DEM skyline database, so that each image obtained a candidate list of DEM location points according to the matching

score. Similar to the method described by the authors of [18], we measured the performance of several localization methods by counting the proportion of the Top_n items in the candidate list containing the correct location. Figure 28 shows that the new algorithm had better performance.

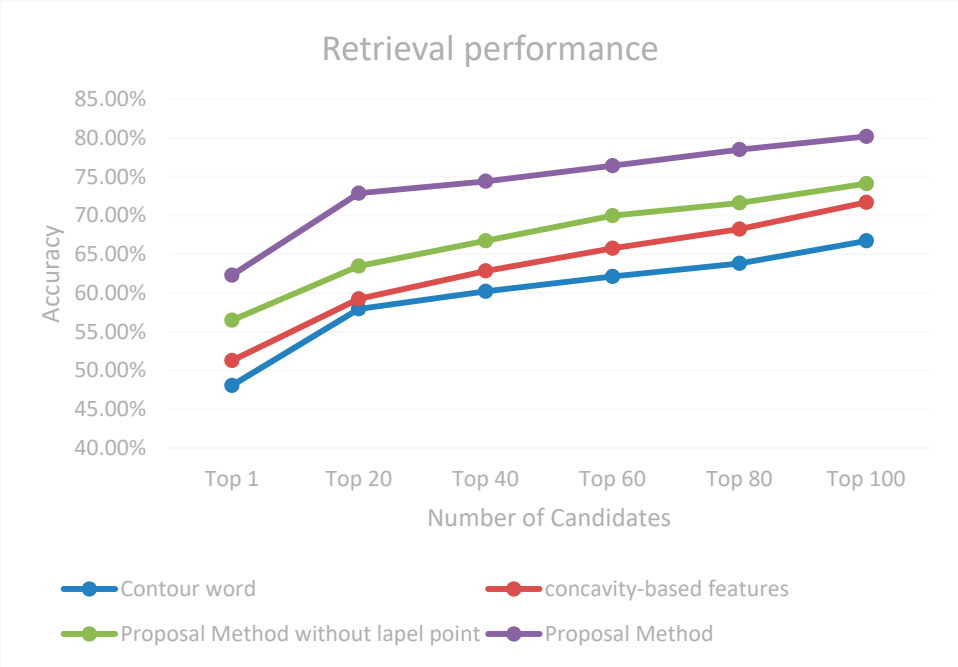


Figure 28. The performance of localization.

Although the proposed method performed better, the method failed on some images with a skyline which was only tens of meters away from the camera (as illustrated in Figure 29).

Figure 29. Examples of failing to locate when the skyline is too close to the camera.

5.3. Robustness against Noise

In order to verify that the proposed method has better localization effect in the case of vegetation, the experiment added noise to photo skylines by randomly adding 1~5 pixels of noise on each image skyline point (as shown in Table 6). Then, the robustness of the two algorithms was calculated. Figure 30 shows the effects of added noise on a skyline. Figure 31 shows that the proposed method and concavity-based method have stronger anti-noise ability.

Table 6. The method of increasing the noise of a skyline (using Python).

The Method of Increasing the Noise of Skyline
for i in range(len(skyline)): noise = random(0,maxNoise [1]) skyline[i] = skyline[i] + noise

[1] maxNoise is equal to 1~5 in turn.

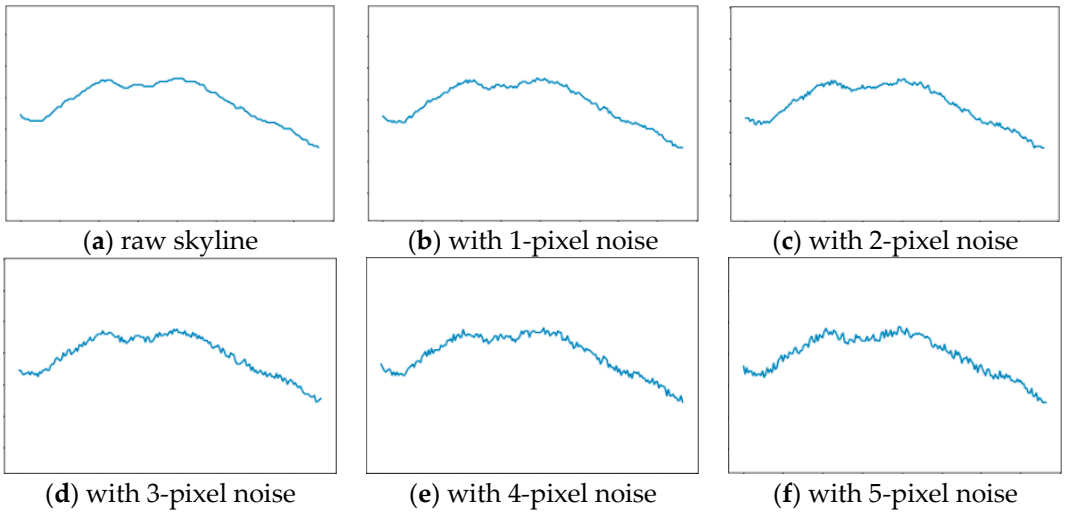


Figure 30. The effect of added noise on a skyline.

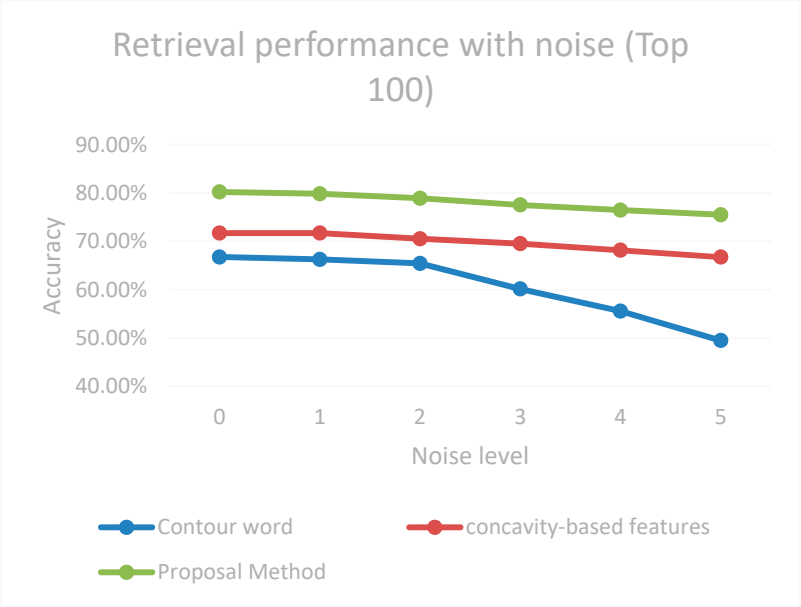


Figure 31. The retrieval performance of two methods with different levels of noise.

5.4. Rotation Invariance

In the experiment, in order to test the effect of camera roll angle on skyline geolocation, the image skyline was rotated by 1~4° in turn along the Z axis, and the matching skyline in the DEM database with the rotated curve was found. This experiment determined the robustness of the proposed method when a skyline was subjected to rotation (see Figure 32).

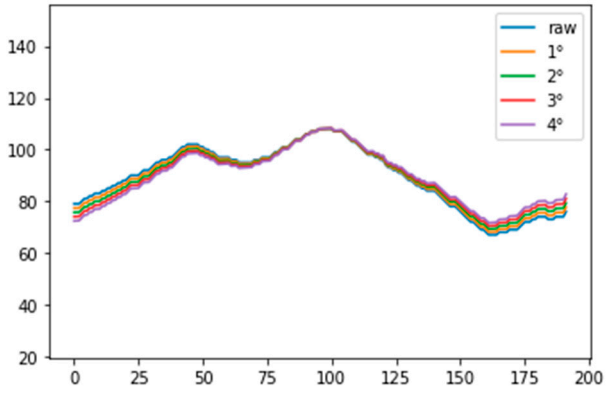


Figure 32. The effect of rotation on a skyline.

Figure 33 shows that the proposal method has a strong adaptability to rotation, and a rotation of 4° had few effects on its accuracy. But with the increase in the degree of rotation, the accuracy of contour word decreased rapidly.

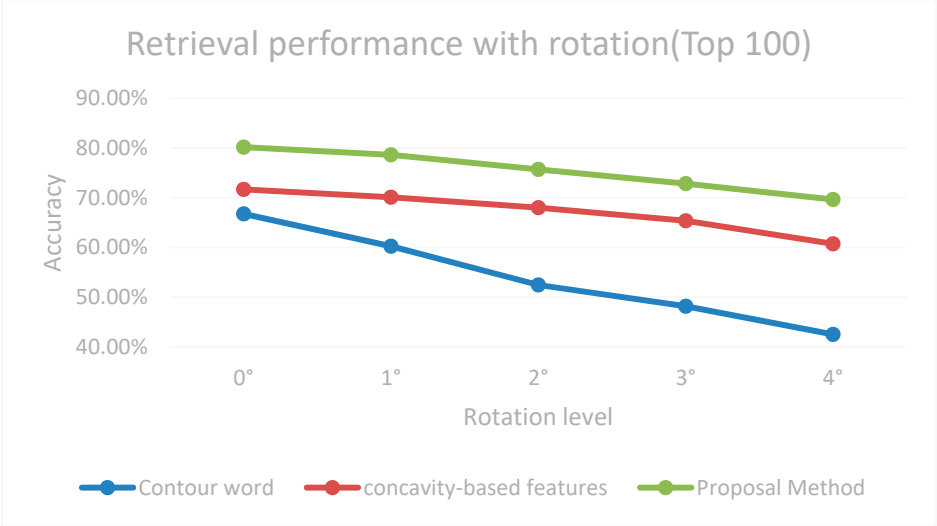


Figure 33. The performance of different rotations.

6. Conclusions and Future Work

Skyline localization is an important branch of visual geolocation, especially in the field where the feature points are sparse. When a GPS signal is unavailable or disturbed, skyline localization is an important auxiliary localization method in the field environment, but there is little research on skyline localization in hilly areas. This paper analyzed the difficulties of skyline locating in hilly areas, including lush vegetation and closeness to skyline. Therefore, a new method of skyline localization was proposed in this paper, which includes two new skyline features: Enhanced Angle chain code and skyline lapel point, and a new matching method which incorporates the skyline pyramid and the skyline distance heatmap. The experimental results show that this method has high localization accuracy in hilly areas. The proposed method also has some limitations. When the nearby trees significantly block the skyline, or when the mountain in front of the camera is too close, the accuracy of the method is low.

In the future, we will investigate enhanced lapel point, which adds the ridge line lapel point (the lapel point formed between ridge lines) to existing skyline lapels, so as to improve the accuracy of geolocation. Although the interference of atmospheric environment is not within the scope of this paper, the skyline geolocation under multiple visibilities is important, because the photo will change due to haze and other reasons.

Author Contributions: J.T., X.X. and Z.W. contributed to the conception of the study; Z.P. performed the experiment; Z.P. and J.T. contributed significantly to analysis and manuscript preparation; Z.P., J.T. and T.T. performed the data analyses and wrote the manuscript; Z.P. and J.T. helped perform the analysis with constructive discussions. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: Conflict of interest Authors Zhibin Pan, Jin Tang, Tardi Tjahjadi, Xiaoming Xiao and Zhihu Wu declare that they have no conflict of interest.

Appendix A

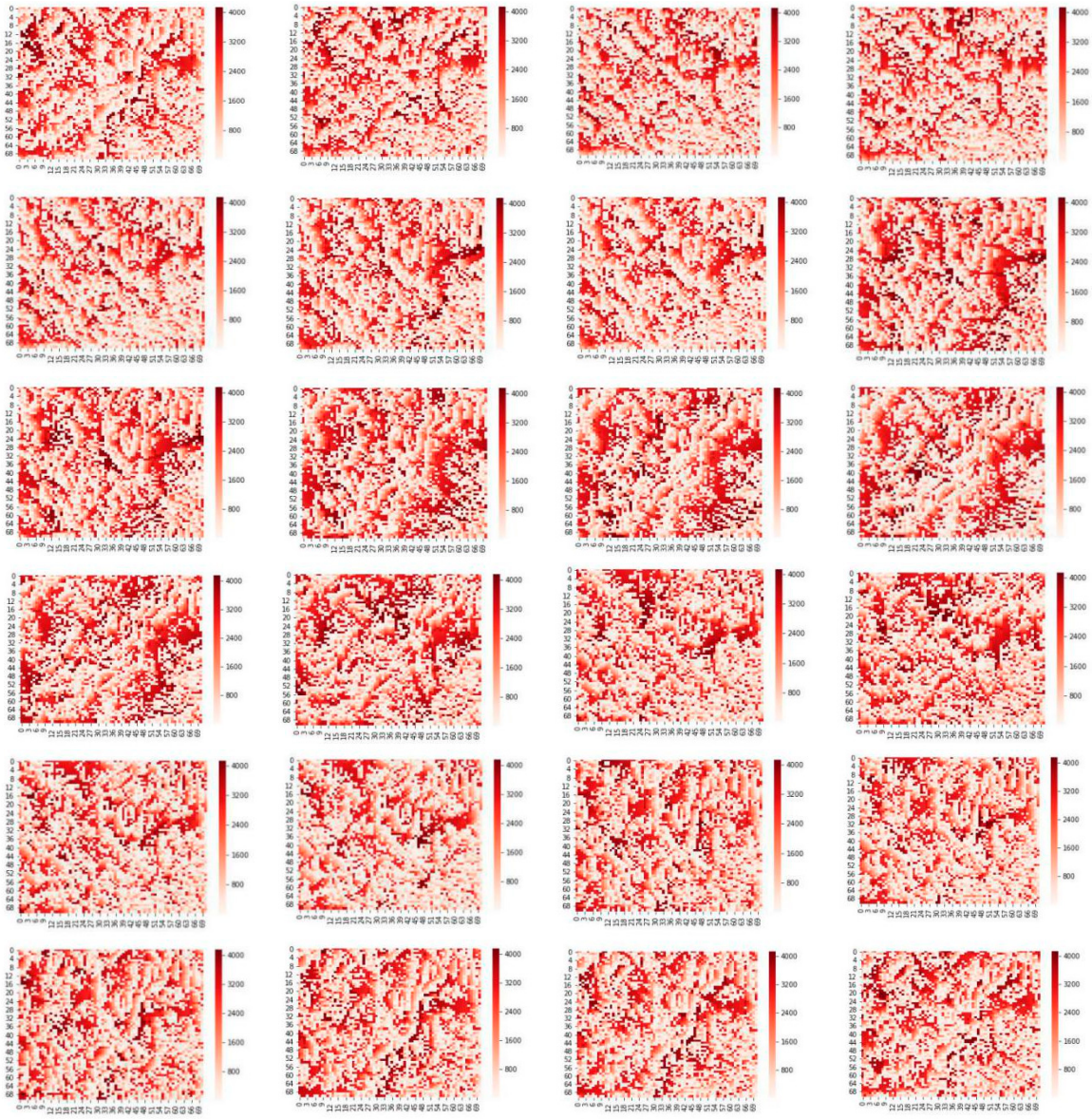


Figure A1. The heatmaps of 24 subdatasets.

References

1. Brejcha, J.; Cadik, M. State-of-the-art in visual geo-localization. *Pattern Anal. Appl.* **2017**, *20*, 613–637. [CrossRef]
2. Gibbens, P.W.; Dumble, S.J. Efficient Terrain-Aided Visual Horizon Based Attitude Estimation and Localization. *J. Intell. Robot. Syst.* **2015**, *78*, 205–221.
3. Ahmad, T.; Bebis, G.; Nicolescu, M.; Nefian, A.V.; Fong, T. An edge-less approach to horizon line detection. In Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 9–11 December 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1095–1102.
4. Porzi, L.; Rota Bulò, S.; Ricci, E. A deeply-supervised deconvolutional network for horizon line detection. In Proceedings of the 24th ACM International Conference on Multimedia, ACM, Amsterdam, The Netherlands, 15–19 October 2016; pp. 137–141.
5. Verbickas, R.; Whitehead, A. Sky and ground detection using convolutional neural networks. In Proceedings of the International Conference on Machine Vision and Machine Learning (MVML), Prague, Czech Republic, 14–15 August 2014.
6. Gupta, V.; Brennan, S. Terrain-based vehicle orientation estimation combining vision and inertial measurements. *J. Field Robot.* **2008**, *25*, 181–202. [CrossRef]
7. Ho, N.; Chakravarty, J. Localization on freeways using the horizon line signature. In Proceedings of the International Conference on Robotics and Automation, Hong Kong, China, 31 May–5 June 2014; pp. 1–8.
8. Dumble, S.J.; Gibbens, P.W. Horizon profile detection for attitude determination. *J. Intell. Robot. Syst.* **2012**, *68*, 339–357. [CrossRef]
9. Neto, A.M.; Victorino, A.C.; Fantoni, I.; Zampieri, D.E. Robust horizon finding algorithm for real-time autonomous navigation based on monocular vision. In Proceedings of the 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), Washington, DC, USA, 5–7 October 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 532–537.
10. Boroujeni, N.S.; Etemad, S.A.; Whitehead, A. Robust horizon detection using segmentation for UAV applications. In Proceedings of the 2012 Ninth Conference on Computer and Robot Vision, Toronto, ON, Canada, 28–30 May 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 346–352.
11. McGee, T.G.; Sengupta, R.; Hedrick, K. Obstacle detection for small autonomous aircraft using sky segmentation. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, 18–22 April 2005; IEEE: Piscataway, NJ, USA, 2005; pp. 4679–4684.
12. Thurrowgood, S.; Soccol, D.; Moore, R.J.D.; Bland, D.; Srinivasan, M.V. A vision based system for attitude estimation of UAVs. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 5725–5730.
13. Grelsson, B.; Felsberg, M.; Isaksson, F. Highly accurate attitude estimation via horizon detection. *J. Field Robot.* **2016**, *33*, 967–993. [CrossRef]
14. Hou, J.; Li, B. An improved algorithm for horizon detection based on OSTU. In Proceedings of the 2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics, Hangzhou, China, 26–27 August 2015; IEEE: Piscataway, NJ, USA, 2015; Volume 1, pp. 414–417.
15. Di, L.; Fromm, T.; Chen, Y.Q. A data fusion system for attitude estimation of low-cost miniature UAVs. *J. Intell. Robot. Syst.* **2012**, *65*, 621–635. [CrossRef]
16. Brejcha, J.; Čadík, M. GeoPose3K: Mountain landscape dataset for camera pose estimation in outdoor environments. *Image Vis. Comput.* **2017**, *66*, 1–14. [CrossRef]
17. Saurer, O.; Baatz, G.; Köser, K.; Ladický, L.; Pollefeys, M. Image Based Geo-localization in the Alps. *Int. J. Comput. Vis.* **2016**, *116*, 213–225. [CrossRef]
18. Brejcha, J.; Cadik, M. Camera Orientation Estimation in Natural Scenes Using Semantic Cues. In Proceedings of the International Conference on 3D Vision, Verona, Italy, 5–8 September 2018; pp. 208–217.
19. Yang, M.; Kidiyo, K.; Joseph, R. A survey of shape feature extraction techniques. *Pattern Recognit.* **2008**, *15*, 43–90.
20. Mokhtarian, F. Multi-scale description of space curves in three-dimensional objects. In Proceedings of the Conference on Computer Vision & Pattern Recognition, Ann Arbor, MI, USA, 5–9 June 1988; Cvpr. IEEE: Piscataway, NJ, USA, 1988.

21. Chen, Y.; Qian, G.; Gunda, K.; Gupta, H.; Shafique, K. Camera geolocation from mountain images. In Proceedings of the International Conference on Information Fusion, Washington, DC, USA, 6–9 July 2015; IEEE: Piscataway, NJ, USA, 2015.
22. Baatz, G.; Irani, M.; Vedaldi, A. Large Scale Visual Geo-Localization of Images in Mountainous Terrain. In Proceedings of the European Conference on Computer Vision, Firenze, Italy, 7–13 October 2012.
23. Sassi, A.; Amar, C.B.; Miguet, S. Skyline-based approach for natural scene identification. In Proceedings of the Computer Systems & Applications, Agadir, Morocco, 29 November–2 December 2016.
24. Tzeng, E.; Zhai, A.; Clements, M.; Townshend, R.; Zakhor, A. User-Driven Geolocation of Untagged Desert Imagery Using Digital Elevation Models. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops, Portland, OR, USA, 23–28 June 2013; pp. 237–244.
25. Freeman, H. On the encoding of arbitrary geometric configurations. *IRE Trans. Electron. Comput.* **1961**, *EC-10*, 260–268. [CrossRef]
26. Arica, N.; Vural, F.T.Y. BAS: A perceptual shape descriptor based on the beam angle statistics. *Pattern Recognit. Lett.* **2008**, *24*, 1627–1639. [CrossRef]
27. Ahmad, T.; Bebis, G.; Nicolescu, M.; Nefian, A.; Fong, T. Horizon line detection using supervised learning and edge cues. *Comput. Vis. Image Underst.* **2020**, *191*, 102879. [CrossRef]