

## Article

# An Optical Remote Sensing Image Matching Method Based on the Simple and Stable Feature Database

Zilu Zhao <sup>1,2,3</sup>, Hui Long <sup>1,2,3,\*</sup> and Hongjian You <sup>1,2,3</sup><sup>1</sup> Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China<sup>2</sup> Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Chinese Academy of Sciences, Beijing 100190, China<sup>3</sup> School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 101408, China

\* Correspondence: longhui@aircas.ac.cn

**Abstract:** Satellite remote sensing has entered the era of big data due to the increase in the number of remote sensing satellites and imaging modes. This presents significant challenges for the processing of remote sensing systems and will result in extremely high real-time data processing requirements. The effective and reliable geometric positioning of remote sensing images is the foundation of remote sensing applications. In this paper, we propose an optical remote sensing image matching method based on a simple stable feature database. This method entails building the stable feature database, extracting local invariant features that are comparatively stable from remote sensing images using an iterative matching strategy, and storing useful information about the features. Without reference images, the feature database-based matching approach potentially saves storage space for reference data while increasing image processing speed. To evaluate the performance of the feature database matching method, we train the feature database with various local invariant feature algorithms on different time phases of Gaofen-2 (GF-2) images. Furthermore, we carried out matching comparison experiments with various satellite images to confirm the viability and stability of the feature database-based matching method. In comparison with direct matching using the classical feature algorithm, the feature database-based matching method in this paper can essentially improve the correct rate of feature point matching by more than 30% and reduce the matching time by more than 40%. This method improves the accuracy and timeliness of image matching, potentially solves the problem of large storage space occupied by the reference data, and has great potential for fast matching of optical remote sensing images.

**Keywords:** image matching; feature database; iterative matching; local invariant features



**Citation:** Zhao, Z.; Long, H.; You, H. An Optical Remote Sensing Image Matching Method Based on the Simple and Stable Feature Database. *Appl. Sci.* **2023**, *13*, 4632. <https://doi.org/10.3390/app13074632>

Academic Editor: Andrés Márquez

Received: 28 February 2023

Revised: 29 March 2023

Accepted: 3 April 2023

Published: 6 April 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

With an increase in satellites and imaging modes that can swiftly and efficiently acquire remote sensing images, the amount of satellite remote sensing image data is exploding. However, the enormous size of the image data has created numerous challenges for traditional remote sensing processing systems, and the processing efficiency of ground-based systems lags behind the satellite observation supply [1]. Although remote sensing processing systems are becoming more intelligent and automated, interactions with satellite systems still take a long time. Traditional image processing systems for remote sensing data are hardly capable of processing instantaneous image applications, making it difficult to fully use the extensive remote sensing image resources [2]. The geometric positioning of remote sensing images plays an important role in remote sensing image applications such as image fusion [3–5], change detection [6–8], image mosaicing [9–11], etc. The initial step and the biggest challenge is image matching. Geometrically precise positioning of remote sensing images is highly dependent on effective and quick image matching.

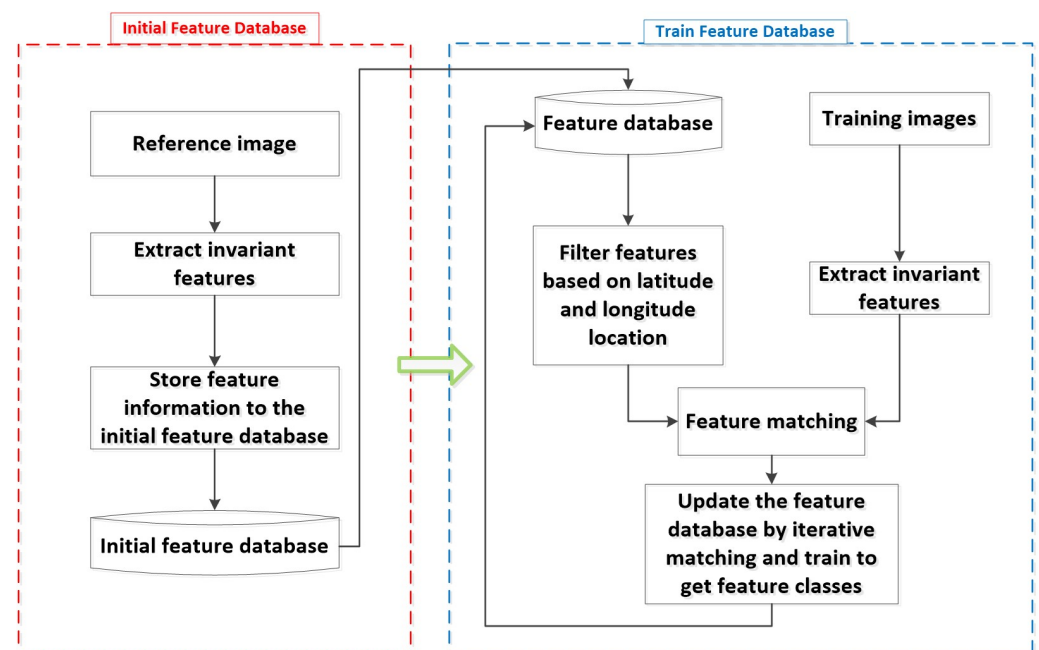
The geometric positioning of remote sensing images requires a large amount of ground control information which is the Ground Control Point (GCP) to establish the transformation relationship between images to achieve the consistency of geometric positioning among images. The traditional method is to analyze the selected GCPs in the image manually, which is time-consuming and laborious, and frequently collects GCPs multiple times. The method of establishing a control point database can be used to retrieve control point image slices in the control point database to achieve quick matching of control points [12]. In addition, when geometric correction is performed on future remote sensing images, existing control points can be searched directly from the control point database to match the images, thus enabling the reuse of control points. However, the matching algorithm of the control points and images is challenging. The control points that are easy to recognize by human eyes are not always easy to recognize by computers, and the content of the control points in the control point database needs to be greatly accumulated to ensure matching to remote sensing images of various time phases. With the development of various local invariant feature extraction algorithms, features can be extracted from the reference image as control point information and used to establish the deformation model between images through feature matching to achieve geometric registration for images [13]. The matching efficiency between remote sensing images is increased by the ability to automatically extract control points created by local invariant features without manual acquisition and storage.

The development of local invariant feature-matching methods has diversified in recent years, which can mainly be divided into the improvement of local invariant features and the improvement of matching strategies. The main goal of developing feature-matching methods for local invariant features is to reduce the complexity of feature algorithms and increase the robustness and discrimination of feature extraction algorithms, such as the scale-invariant feature transform (SIFT) [14] and its subsequent variants principal components analysis SIFT (PCA-SIFT) [15], gradient location and orientation histogram (GLOH) [16], the speeded up robust feature (SURF) [17], the uniform robust SIFT (UR-SIFT) [18], KAZE [19], accelerated-KAZE (AKAZE) [20], etc. To further increase the effectiveness of the feature matching algorithm, techniques combining accelerated detectors with binary descriptors, such as the oriented FAST and rotated BRIEF (ORB) [21], the binary robust invariant scalable keypoints (BRISK) [22], and the fast retina keypoint (FREAK) [23] algorithms, have emerged. These techniques are not only quick at matching but also have exceptionally low storage space requirements, which helps resolve the issue with real-time matching methods. For matching strategies, feature matching techniques are improved, and various techniques are used to increase the accuracy of feature extraction on remote sensing images. For example, Ma et al. proposed a novel remote sensing image feature matching error removal method, introducing a guided matching strategy to significantly improve the number of true matches without sacrificing accuracy [24]. Li et al. proposed the adaptive regional multiple feature matching method (ARMF), which introduces an adaptive region search strategy to adaptively select matching feature regions using pyramidal scaling techniques to extract multiple types of features and adaptively select appropriate feature descriptors [25]. Chen et al. proposed a closed-feedback SIFT system based on SIFT, containing a correction return loop to improve the position accuracy by replacing the current sensed image via an iterative approach [26].

The key to the above-mentioned local invariant feature-based matching approaches lies in pairing highly-repeatable common features between the same scene images under multiple observation conditions [27]. These features are invariant to different images of the same scene, should be unique within the same image, and can be distinguished from other feature points within the image using descriptors [28]. However, many existing local invariant feature-matching methods have problems in the following aspects. Firstly, it is difficult for existing algorithms to ensure that the extracted local invariant features remain invariant in multi-temporal, multi-radiation, and multi-view remote sensing image data. Because many feature extraction methods are based on grayscale images or gradient information to detect keypoints that are easily affected by noise and illumination, leading to

wrong detection and omission, and the descriptors cannot express features correctly, under different observation settings, there will not be enough feature point correspondences to guarantee the accuracy of image matching [29]. Secondly, these feature-matching methods require accurate geographic information from reference images. The acquisition and selection of reference images restrict remote sensing image matching under multiple observation conditions, since it is challenging to guarantee effective matching with a single reference image, which is less flexible. Additionally, with the rapid development of remote sensing imaging technology, the resolution and size of the collected multi-observation condition images have been greatly increased, and the storage space occupied has increased. This makes reading remote sensing images and extracting features from remote sensing images less efficient.

In response to the above problems, this paper proposes a fast-matching method for optical remote sensing images based on simple and stable feature database matching. First, considering that the feature-based matching method is constrained by the selection of the reference image, the feature database is applied to the remote sensing image matching method. The feature database refers to the control point database production method and stores the local invariant features in a simple and effective form. These features can be directly applied in the subsequent geometric positioning, saving the time of extracting features in the reference image and improving the image matching efficiency. Secondly, this paper combines the training-feedback mechanism to iteratively match the feature database, construct stable feature sets, and cluster the descriptors of stable feature points under multi-observation conditions. After the feature database is trained, multiple relatively stable feature point sets are stored in the database. Finally, the test images are matched with the feature database to achieve fast and accurate image geometry correction. In this study, we create stable feature databases using a variety of local invariant features and perform feature database matching experiments using a data set in the western Beijing region. Figure 1 shows the main flow of the proposed method.



**Figure 1.** Main flow of the construction of the feature database.

The main contributions of this paper are as follows:

1. Imitating the control point database to create the feature database. Compared with a reference image or control point database, the feature database uses less storage space and takes a shorter time to match with remote sensing images.

2. A training feedback feature database iterative matching strategy is proposed. Unlike analyzing the robustness of features by methods such as information entropy or feature metric, this method analyzes the stability of features in different temporal training images longitudinally, constructs stable feature point sets, and improves the correct matching rate of optical image feature points and image matching success rate under multi-observation conditions.

The remainder of this paper is organized as follows. Section 2 first briefly reviews the classical feature algorithms required to build feature databases and then describes the proposed feature database iterative matching strategy. In Section 3, the experimental results of the feature database matching approach are shown and the superiority of the feature database approach is demonstrated compared to direct matching using classical algorithms. In Section 4, our proposed feature database matching approach and suggestions for future work are discussed. Conclusions are drawn in Section 5.

## 2. Materials and Methods

In this section, we first discuss the local invariant feature extraction algorithms—SIFT, SURF, KAZE, AKAZE, ORB, and FREAK—that were used to build the feature database and investigate the methods for extracting and describing these features. Table 1 provides a quick comparison of the feature algorithms. The creation of a simple stable feature database and iterative training is then detailed.

### 2.1. Common Local Invariant Features

SIFT, SURF, KAZE, AKAZE, ORB, and FREAK are selected in the paper for the construction of feature databases separately. These six local invariant features are selected because they are typical scale invariant feature detection methods widely used in image matching and remote image registration. Basically, they detect significant keypoints in the image and obtain descriptors that correspond to the keypoints individually. The four main steps of these algorithms are building scale space, detecting key points, assigning feature orientation, and constructing descriptors. These six methods are briefly described in Table 1.

The feature database contains information primarily as feature point longitude, latitude, response, dominant orientation, descriptors, etc. Table 2 shows the features information stored in the feature database of six feature algorithms. The traditional GCP storage mode requires not only coordinates of the control point but also a local image centered on the control point. The size of the control point local image is generally between  $100 \times 100$  pixels and  $200 \times 200$  pixels in order to contain obvious features. In the case where the resolution of the remote sensing image is high, a larger-scale local image of the control point will be required. Table 2 compares the information on the features in the feature database with the information on conventional control points, and it is clear that the feature database is superior to the control point database in terms of storage space, greatly compressing the reference data storage space, especially for the binary descriptors of the ORB which are compressed by a factor of up to 500 or more.

### 2.2. The Building Method of Simple Stable Feature Database

This section builds simple stable feature databases using the various local invariant features that were introduced in the previous section. A simple stable feature database is created in two steps: first, relevant features are extracted from a reference image that is geographically accurate, and their useful information is stored in the initial feature database. Next, training images are continuously matched with the feature database, and the content of the feature database is updated based on the matching results, such as by adding and removing feature points. This feature database construction process can be seen in Algorithm 1.

In the process of continuous matching iterations, relatively stable invariant feature points are obtained. The results of the test image registration depend heavily on the pre-

training of the feature database. Because of this, we pre-align the input remote sensing training images to make sure that the pixel offset between the reference image and the training images is not too large. We also geometrically correct the training image during the training matching process to make sure that the feature points in the feature database and the reference image are geometrically consistent.

**Table 1.** Brief introduction of multiple local invariant feature algorithms.

Algorithm	Detector Type	Description
SIFT	Blobs	SIFT first constructs the Difference-of-Gaussian scale space based on the Gaussian scale space; next, the keypoints are detected and precisely located in the Difference-of-Gaussian images; after that, one or more orientations are determined by the peak of the gradient histogram of each key point neighborhood. Finally, 128-dimensional feature descriptors are constructed based on the gradient information of the neighborhood centered on the keypoints.
SURF	Blobs	SURF uses box filters to convolve with the original image to construct the scale space while using the integral image technique to increase the computational efficiency of the algorithm. It detects the candidate feature points using the Hessian Matrix, followed by non-maximal suppression. To determine the keypoint orientation, SURF adds up the Haar-wavelet responses in the horizontal and vertical orientation of the $\frac{\pi}{3}$ angular sector sliding window in the circular neighborhood of the keypoint. The orientation of the longest such vector sum is used as the keypoint orientation. Furthermore, SURF statistics the Haar-wavelet responses in the area around the keypoint to create a 64-dimensional feature descriptor.
KAZE	Blobs	KAZE uses efficient Additive Operator Splitting (AOS) techniques for nonlinear diffusion filtering to build nonlinear scale space, which reduces noise while maintaining edges. It searches for Hessian local maxima on the nonlinear scale space after normalization at different scales as the keypoint and finds the dominant orientations of feature points in a similar way to SURF. KAZE builds the descriptor using a variant of the SURF descriptor, Modified-SURF (M-SURF) [30], which can handle the boundaries better than the original SURF descriptor and finally forms a 64-dimensional feature vector.
AKAZE	Blobs	AKAZE is an accelerated variant of KAZE. It constructs nonlinear scale spaces more quickly by using the Fast Explicit Diffusion (FED) mathematical framework. Similar to KAZE, It locates candidate points and filters them at each octave to perform keypoint extraction. It calculates the dominant orientations of keypoints in a similar way to KAZE. It uses an updated Modified-Local Difference Binary (M-LDB) binary descriptor for descriptor construction that not only compares region means instead of individual pixels in the binary set but also incorporates rotation invariance.
ORB	Corners	ORB consists of a modified FAST (Features from Accelerated Segment Test) [31] and a rotated BRIEF (Binary Robust Independent Elementary Features) [32]. It first uses FAST to quickly select keypoints; afterward, it uses the intensity centroid method to calculate keypoint orientations; and finally, it uses the modified BRIEF to create binary descriptors, compared with the original BRIEF descriptors with increased rotational invariance compared with the original BRIEF.
FREAK	Corners	Only the descriptor extraction approach is improved by FREAK. The keypoint detection algorithm in BRISK is used in the original paper to perform FAST feature point detection on the constructed multi-scale space. However, unlike the uniform sampling pattern of BRISK for extracting feature descriptors, FREAK is inspired by the human visual system and uses a retinal sampling pattern where the smaller the distance from the keypoint, the denser the sampling, and the larger the distance from the keypoint, the more discrete the sampling points.

**Table 2.** Comparison of single control point storage and single feature points extracted by different feature extraction algorithms.

Control Point Mode	Storage Content	Storage Type	Storage Size/Byte	Total Storage Size/Byte	Compression Ratio
Control Point	Longitude and Latitude	float	8	40,008	
	Local Image (200 × 200)	unsigned char	40,000		

Table 2. Cont.

Control Point Mode	Storage Content	Storage Type	Storage Size/Byte	Total Storage Size/Byte	Compression Ratio
SIFT	128-dimensional Descriptor	float	512	556	1/72
	Point properties <sup>1</sup> , Update parameters <sup>2</sup>	float	44		
SURF	64-dimensional Descriptor	float	256	300	1/133
	Point properties <sup>1</sup> , Update parameters <sup>2</sup>	float	44		
KAZE	64-dimensional Descriptor	float	256	300	1/133
	Point properties <sup>1</sup> , Update parameters <sup>2</sup>	float	44		
AKAZE	64-dimensional Descriptor	unsigned char	64	108	1/370
	Point properties <sup>1</sup> , Update parameters <sup>2</sup>	float	44		
ORB	32-dimensional Descriptor	unsigned char	32	76	1/526
	Point properties <sup>1</sup> , Update parameters <sup>2</sup>	float	44		
FREAK	64-dimensional Descriptor	unsigned char	64	108	1/370
	Point properties <sup>1</sup> , Update parameters <sup>2</sup>	float	44		

<sup>1</sup> Feature point latitude, longitude, response, angle, size, octave. <sup>2</sup> Number of matches, number of unmatched matches, number of consecutive matches, number of consecutive unmatched matches, feature class label.

### 2.2.1. Initial Feature Database Building

Adaptive histogram equalization is first used to pre-process the reference image  $R$ , and then the chosen feature algorithm is used to extract the feature points. The *Longitude*, *Latitude*, *Octave*, *Response*, *Angle*, *Size*, number of matches ( $M$ ), number of unmatched matches ( $UM$ ), number of consecutive matches ( $CM$ ), and number of consecutive unmatched matches ( $CUM$ ) are recorded for each feature, along with the feature descriptor *Descriptor*. Each feature in the initial feature database has an *Index*, which is used to group similar features to form feature classes. Then, all the collected features are assembled as  $F_0 = \{f_1^0, f_2^0, \dots, f_{n_0}^0\}$  written in the initial feature database where  $f_i^k$  represents the feature class whose *Index* is  $i$  in the feature database,  $k$  is the number of images input and  $n_0$  is the total number of feature classes in the feature set. At first,  $n_0 = N_0$ , the matching parameters of each feature in the initial feature database are  $M_i = CM_i = UM_i = CUM_i = 0$ ,  $i = 0, 1, 2, \dots, n_0$ , and  $N_0$  is the total number of all feature points in the feature set  $F_0$ .

### 2.2.2. Iterative Matching Strategy for Feature Database Methods

The feature database iterative matching strategy can not only train the stable feature points, but also cluster feature descriptors in multiple observation conditions at the same location.

During feature matching, the clustered feature sets (feature classes) filtered by the feature database change the one-to-one case of descriptors at the same location to one-to-many, providing more matching possibilities for the test images. Furthermore, as an iterative process of the feature database, the number of matches of feature classes is recorded. After matching the training image with the feature database, the successfully matched pairs of points are aggregated into feature classes, and parameters such as the number of matches are recorded, while unsuccessful matches are also added to the feature database and the given relevant parameters. Each feature class is assigned a label and the features in the feature class will be continuously added or removed with the matching of results of the subsequent training images. After matching the initial feature database with the first training image, all subsequent training images are matched with the feature class, and features are considered to be successfully matched with the features at that location if they are successfully matched with the feature class.



**Algorithm 1:** Feature Database Construction

---

**Input:** reference image  $R$ , training images  $\{S_1, S_2, \dots, S_N\}$ , the threshold of not matched,  $thre1$ , the threshold of consecutive unmatched,  $thre2$

**Output:** feature database

- 1 Initialize: extract feature points from  $R$ .  $F_0 = \{f_1^0, f_2^0, \dots, f_{n_0}^0\}$  is written to the initial feature database.  $M_i = CM_i = UM_i = CUM_i = 0, i = 0, 1, 2, \dots, n_0$ ;
- 2 **for**  $k = 1, 2, \dots, N$  **do**
- 3     get  $F_k = \{f_1^k, f_2^k, \dots, f_{n_f}^k\}$  from the feature database;
- 4     extract  $T_k = \{t_1^k, t_2^k, \dots, t_{n_t}^k\}$  from  $S_k$ ;
- 5     match  $F_k$  with  $T_k$ ;
- 6     **for**  $j$  in  $F_k$  **do**
- 7         **if**  $j$  is matched **then**
- 8              $M_j = M_j + 1; CM_j = CM_j + 1; UM_j = UM_j; CUM_j = 0$ ;
- 9         **else**
- 10             **if** the feature class  $f$  containing  $j$  is matched **then**
- 11                  $M_j = M_j + 1; CM_j = CM_j + 1$ ;
- 12                  $UM_j = UM_j + 1; CUM_j = CUM_j + 1$ ;
- 13             **else**
- 14                  $M_j = M_j; CM_j = 0; UM_j = UM_j + 1; CUM_j = CUM_j + 1$ ;
- 15             **end**
- 16         **end**
- 17     **end**
- 18     **for**  $j$  in  $T_k$  **do**
- 19         **if**  $j$  is matched **then**
- 20              $M_j = M_j + 1; CM_j = CM_j + 1; UM_j = 0; CUM_j = 0$ ;
- 21             added  $j$  to  $F_k$  and assign  $j$  the same *Index* as the matched feature class;  
//  $M_j$  and  $CM_j$  are consistent with the matched feature. This will not be repeatedly added here;
- 22         **else**
- 23              $M_j = 0; CM_j = 0; UM_j = 1; CUM_j = 1$ ;
- 24             added  $j$  to  $F_k$  and assign  $j$  a new *Index*;
- 25         **end**
- 26     **end**
- 27     **for**  $j$  in  $F_k$  **do**
- 28         **if**  $\frac{UM_j}{M_j + UM_j} > thre1$  or  $CUM_j > thre2$  **then**
- 29             remove  $j$  from  $F_k$ ;
- 30         **end**
- 31     **end**
- 32     rewrite the updated  $F_k$  to the feature database;
- 33 **end**

---

The feature class that has more matches during the training process indicates that the local features of the scene do not change much in that period. Then, for the images in this period, the probability of the successful matching of feature classes with a higher matching time is greater than that of the successful matching of feature classes with a lower matching time. Figure 2 shows the flow chart of the proposed iterative matching method. The specific iterative matching process is as follows:

- A. Training image feature set extraction: The training image set  $\{S_1, S_2, \dots, S_N\}$  is input in temporal order to match the feature database, where  $N$  is the number of training images in the training set. When the  $k$ th training image  $S_k$  is input, the training image is first pre-processed with adaptive histogram equalization, then the feature

set  $F_k = \{f_1^k, f_2^k, \dots, f_{n_f}^k\}$  is read from the feature database according to the latitude and longitude of the training image, and then the feature set  $T_k = \{t_1^k, t_2^k, \dots, t_{n_t}^k\}$  is extracted from the training image  $S_k$  using the same feature extraction method as the extraction of the reference image features, where  $n_f$  is the number of feature classes in  $F_k$  and  $n_t$  is the number of feature classes in  $T_k$ .

- B. Feature database feature set update: Feature matching is performed in the feature database set  $F_k = \{f_1^k, f_2^k, \dots, f_{n_f}^k\}$  and the training image set  $T_k = \{t_1^k, t_2^k, \dots, t_{n_t}^k\}$ . We use the nearest-neighbor distance ratio (NNDR) [14] method based on descriptor distance to select the correspondence, and then the fast sample consensus (FSC) [33] technique is used to filter error matching [34]. Different distance matching methods are used for different feature algorithms, such as KAZE, SIFT, and SURF, which use Euclidean distance matching, and ORB, FREAK, and AKAZE, which use Hamming distance matching. In this paper, image blocks are matched with the feature database and then the number of matched feature points or feature classes exceeds 10 after FSC filtering is considered stable matching, and the matching results are regarded valid. The following step is a modification of  $F_k = \{f_1^k, f_2^k, \dots, f_{n_f}^k\}$  based on the matching results of  $F_k = \{f_1^k, f_2^k, \dots, f_{n_f}^k\}$  and  $T_k = \{t_1^k, t_2^k, \dots, t_{n_t}^k\}$  on the basis of valid matching.
- The feature points successfully matched in  $F_k$ :  
The matching parameters of the feature  $j$  in  $F_k$  are updated directly, the number of matches is increased by one,  $M_j = M_j + 1$ , the number of consecutive matches is increased by one,  $CM_j = CM_j + 1$ , the number of unmatched matches remains unchanged,  $UM_j = UM_j$ , and the number of consecutive unmatched matches is reduced to zero,  $CUM_j = 0$ .
  - Feature points in  $F_k$  that failed to match:  
The matching parameters are updated according to the label of the unmatched feature  $j$  in  $F_k$ . If there is a successful matching point with the same label *Index* as the feature  $j$ , it means that the feature class of the feature  $j$  is successfully matched, but the feature  $j$  is not successfully matched; at this time, the number of matches is increased by one,  $M_j = M_j + 1$ , the number of consecutive matches is increased by one,  $CM_j = CM_j + 1$ , the number of unmatched points is increased by one,  $UM_j = UM_j + 1$ , and the number of consecutive unmatched points is increased by one,  $CUM_j = CUM_j + 1$ . If there is no successful matching point with the same label, the number of matches remains unchanged,  $M_j = M_j$ , the number of consecutive matches is cleared,  $CM_j = 0$ , the number of unmatched matches increases by one,  $UM_j = UM_j + 1$ , and the number of consecutive unmatched matches is increased by one,  $CUM_j = CUM_j + 1$ .
  - The feature points successfully matched in  $T_k$ :  
In  $T_k$ , the matched feature  $j$  belongs to a feature class of  $F_k$ . After correcting the position according to the matching result, the feature  $j$  is added to  $F_k$  with the same *Index*,  $M_j$  and  $CM_j$  changed to the same as the matched features in  $F_k$ , with zero unmatched,  $UM_j = 0$  and zero consecutive unmatched,  $CUM_j = 0$ .
  - Feature points in  $T_k$  that failed to match:  
The unmatched feature  $j$  in  $T_k$ , after correcting the position according to the matching result, will also be added to the feature database set with a new *Index* for the newly added feature, the number of matches and consecutive matches is 0,  $M_j = CM_j = 0$ , and the number of unmatched and consecutive unmatched is 1,  $UM_j = CUM_j = 1$ , which corresponds to the newly added feature points.
  - For other features that are not in the extracted feature set  $F_k$  but exist in the feature database, they remain unchanged in the feature database.
- C. Delete feature points within the feature set based on the threshold: After the above steps, the features are filtered for  $F_k$ , and when the proportion of unmatched times



between them to the total number of all training at that point exceeds a threshold,  $\frac{UM_j}{M_j+UM_j} > thre1$ , or when the number of consecutive unmatched times exceeds a threshold,  $CUM_j > thre2$ , the features are removed from the feature set  $F_k$ . In this paper, we empirically set the parameters  $thre1 = 0.5$  and  $thre2 = 4$  to rewrite the filtered feature set  $F_k$  into the feature database.

- D. Repeat the above process to iteratively train the feature images: The feature database is continuously updated and iterated using training images, in which feature points that can be matched multiple times are automatically clustered to obtain stable feature classes with the same label, which is equivalent to aggregating multiple feature descriptors at the same location under multiple observation conditions, thus realizing the training process of the feature database.

The feature class is a cluster of descriptors of the same feature point location under different observation conditions. For the same feature class, the number of matches is identical. In this case, the matching number of a feature class represents to some extent the stability of the feature class. A high feature matching number indicates that the feature class appears more times in the training iteration process, while a low feature matching number indicates that the feature class successfully matches fewer times in the training process.

In this paper, the match number of feature classes is used as the standard for filtering stable features. The value of  $FMN$  (filter match number) is used to filter stable features in the feature database. The value of  $FMN$  is a key parameter that determines the number and stability of feature classes obtained from the feature database. For example,  $FMN = 6$  indicates that the feature classes with the number of matches greater than or equal to 6 are extracted from the feature database. The  $FMN$  of the simple stable feature database needs to be determined by a filtering experiment of matching times. The value of  $FMN$  is determined by the observation of the correct matching rate curve in the experiment.  $FMN$  needs to ensure that the feature database has a certain number of feature classes obtained by filtering and yet can be matched with images to obtain a high correct matching rate.

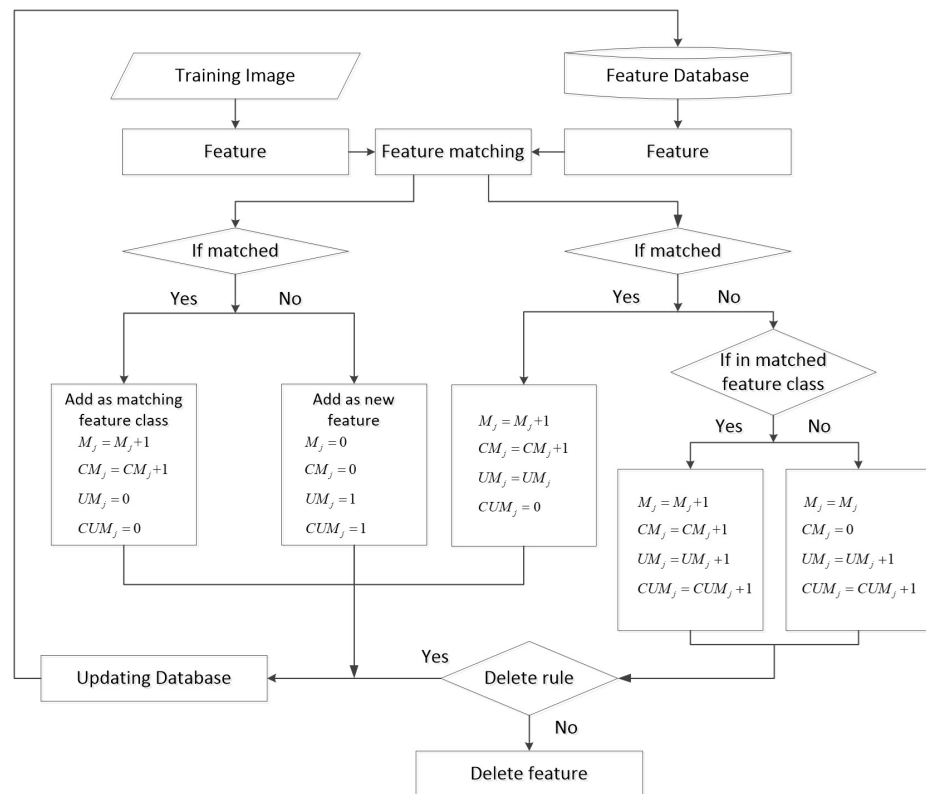


Figure 2. Flow chart of feature database training.

### 3. Experiments and Results

In this section, we build several feature databases based on different feature extraction algorithms and train the feature databases according to the method proposed in Section 2.2. We first set up different *FMN* filtering feature classes from the feature database and evaluate the performance of the proposed feature database matching method by the matching effect of the filtering results with the test images. Then, the feature database matching methods based on different feature algorithms are compared with the reference image matching methods to verify the efficiency and accuracy of the feature database matching methods.

#### 3.1. Evaluation Indicators

Three quantitative metrics are employed in this experiment section to evaluate how the proposed feature database matching method performs.

1. The stability of the feature matching method is measured by the correct matching ratio (*CMR*). It is the proportion of the number of correct matches (*NCM*) to the total number of feature matches (*Nt*). Because a feature class in the feature database matching method contains multiple similar features and a successful match of one of the features means a successful match of this feature class, the calculation of the *CMR* matching method of the feature database is based on the feature classes described in this paper.

$$CMR = \frac{NCM}{Nt} \quad (1)$$

2. Root mean square error (*RMSE*), which is used to reflect the geometric localization accuracy of the feature matching method, where  $(x_i^r, y_i^r)$  denotes the coordinates of the matched feature points in the reference image or feature database and  $(x_i^t, y_i^t)$  denotes the corresponding coordinates of the matched points in the test image after geometric correction. A smaller *RMSE* denotes a higher degree of geometric localization accuracy using the feature-matching approach.

$$RMSE = \sqrt{\frac{1}{NCM} \sum_{i=1}^{NCM} ((x_i^r - x_i^t)^2 + (y_i^r - y_i^t)^2)} \quad (2)$$

3. The total time (*Time*) spent to extract features and perform feature matching from the reference image or feature database and the test image, which is used to reflect the matching efficiency of the feature matching method.

#### 3.2. Experimental Data

To test the feature database matching method, a data set is used in the experiments.

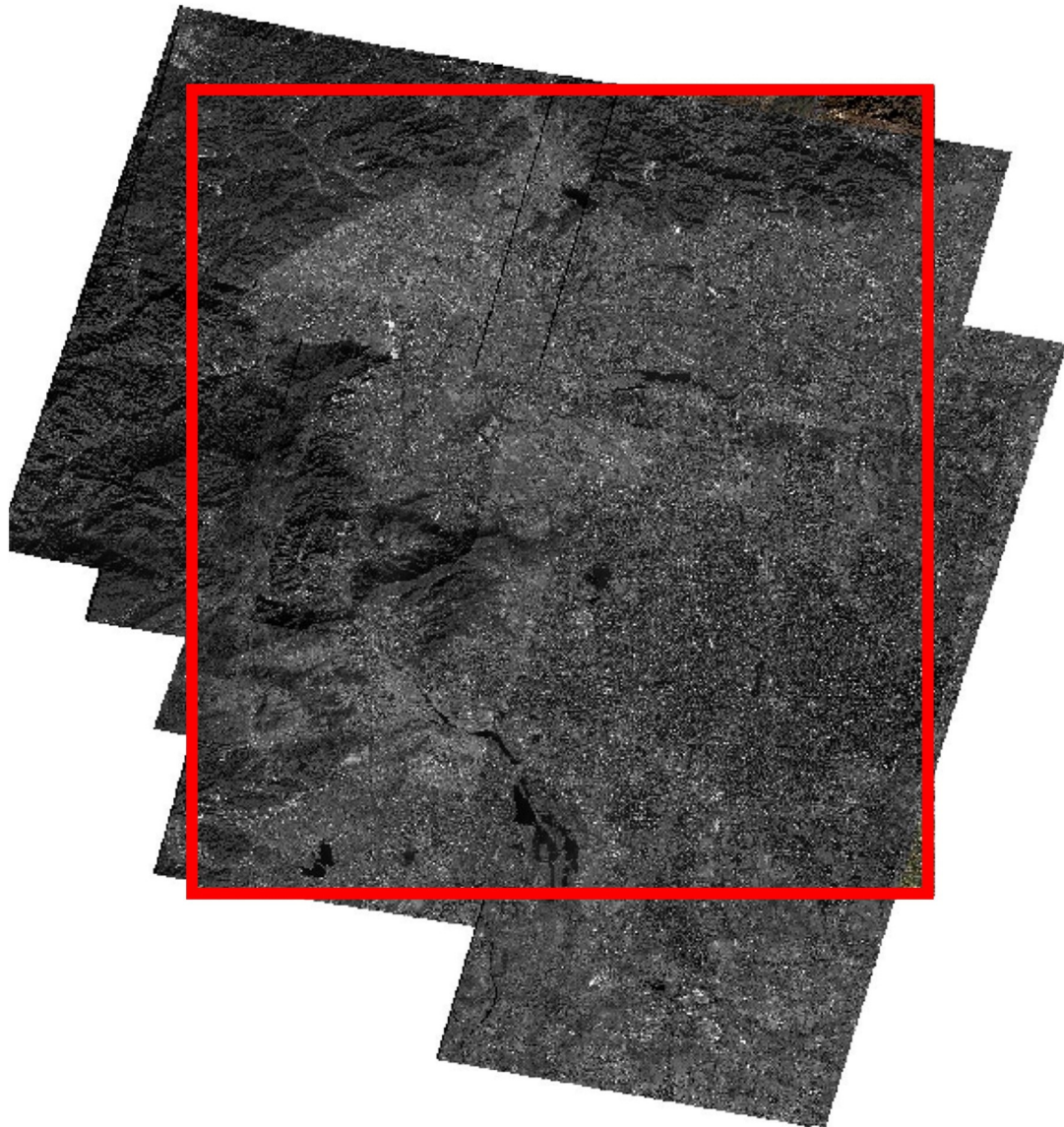
The data set takes the western area of Beijing as the experimental area and contains one reference image, fifty training images, and nine test images. The reference image is a large-scale Google image with an accurate geographic location, and the image range is the latitude and longitude range of the feature database. The training set contains 50 high-resolution remote sensing images of Gaofen-2 (GF-2), these remote sensing images are basically within the set feature database area, overlapping each other, and are used for feature database iterative training. The test set is selected from three images of Jilin-1 (JL-1), Gaofen-1 (GF-1), and GF-2 remote sensing images, respectively, which are used to match with the trained feature database. The latitude and longitude of the test images are also basically within the matching range of the feature database.

To ensure the training of the feature database and subsequent geometric localization, we pre-aligned the training images in the two input remote sensing datasets. The interpolation algorithm first unifies the multiple-resolution images in this paper into a single resolution before matching and comparing them. Because remote sensing images can be based on a uniform scale of geographic information, matching on the same scale can improve the correct rate of matching and avoid incorrect matching due to different scales.

The experiments require training feature databases and matching test images for multiple feature algorithms. For the consideration of computer performance and experiment time, the data set D has been unified to the resolution of 4m, which can accelerate the experiment process while maintaining a relatively high resolution. The details of the data set, including the image source, number of images, size, acquisition time, and spatial resolution, are listed in Table 3 and shown in Figure 3. The red box in Figure 3 represents the range of the reference image.

**Table 3.** Information for the data set.

Image	Source	Number	Date	Size (Pixel × Pixel)	Resolution (m)
Reference (A)	Google Earth	1	2016	53,120 × 49,152	1.19
Training (B)	GF2	50	2016–2022	27,620 × 29,200	0.81
	JL1 (C-1 C-2 C-3)	3	2019–2020	28,651 × 28,720	0.75
Test (C)	GF1 (C-4 C-5 C-6)	3	2019–2021	18,236 × 18,190	2
	GF2 (C-7 C-8 C-9)	3	2019–2021	27,620 × 29,200	0.81



**Figure 3.** Diagram of the data set.

### 3.3. Large Area Remote Sensing Image Feature Database Matching Experiment

#### 3.3.1. Matching Times Filtering Experiment

First, the feature database update parameters  $thre1 = 0.5$ ,  $thre2 = 4$  are set empirically, and several feature databases based on different feature operators are trained according to the method described in Section 2.2 in combination with the reference image and training set. To ensure the uniformity of matched feature points in large regions, a block processing strategy is adopted in the feature database feature extraction, training process, and the subsequent matching process with the test set. The training process ensures that different feature algorithms extract the same number of features for the same training image for subsequent experimental comparison.

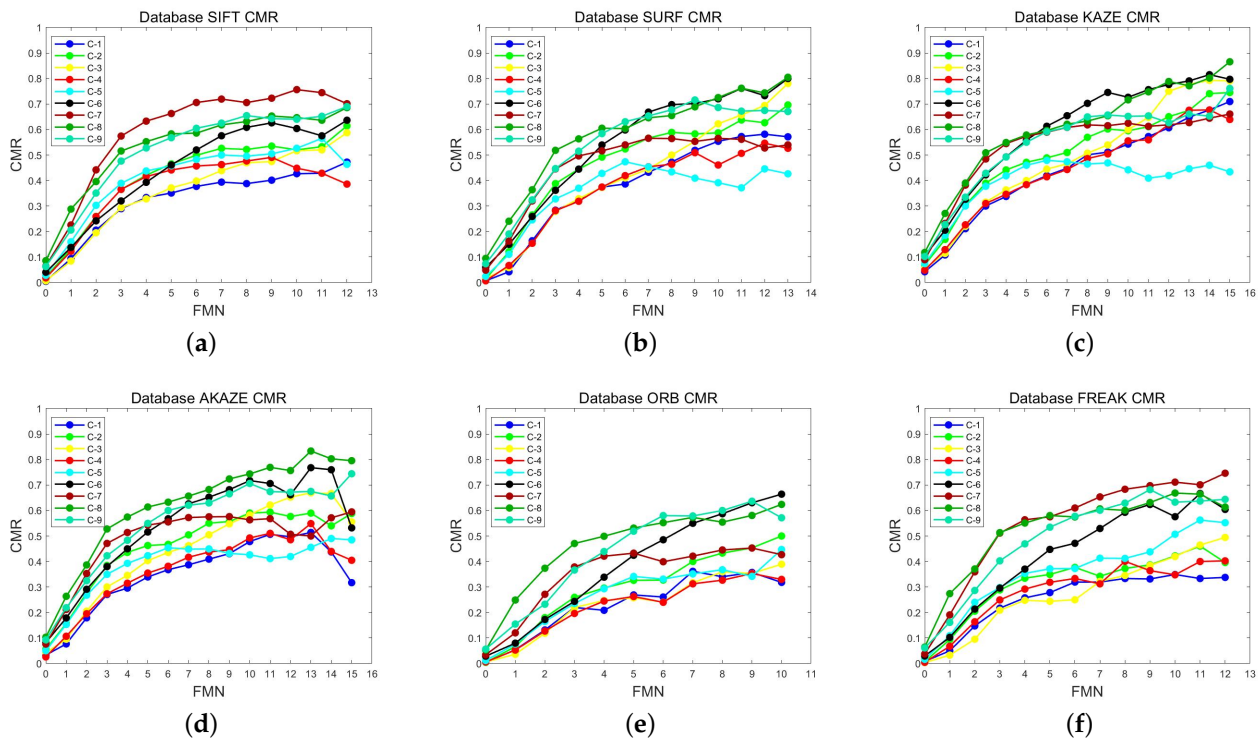
After obtaining several feature databases based on the training set  $B$  and different feature extraction algorithms, the test set  $C$  is matched with the stable feature classes extracted from these feature databases using the  $FMN$ . There are a total of 50 training images available, and the ideal situation is that there are enough feature classes with a matching count of 50, that is, multiple feature classes are correctly matched in each training image.

However, it is impossible to achieve the ideal situation in the actual experimental process. First of all, the latitude and longitude ranges of the 50 training images are not exactly the same, they overlap with each other, and there are even cases of complete non-overlap, so it is impossible to match the features extracted from the reference image with all the 50 training images. Moreover, the number of filtered feature classes decreased with increasing  $FMN$ . Since not all extracted feature points can be matched accurately, the matching process will miss the correct match, and it is not guaranteed that all similar feature points in the training image can be stored in a single feature class with the same feature label. In addition, the training image's features are impacted by the passage of time because it was taken at a different time than the one to which it belongs. There will undoubtedly be new features during the training image iteration, and feature points that cannot be successfully matched will be eliminated. Therefore, the number of feature classes with more matches is smaller, which also suggests that the feature class is more stable in this region range.

Because different feature extraction algorithms exhibit different stability during iterative training of the feature database, the maximum  $FMN$  value is determined by the number of feature classes that can be filtered from the feature database within the test image region. The experiments in this section set a minimum threshold of 50 to ensure that a sufficient number of feature classes are extracted from the test image. As a result, the set  $FMN$  is only added to the statistics when the number of feature classes extracted from the feature database in accordance with  $FMN$  is greater than or equal to 50. As can be seen in Figure 4, the difference in the stability of the feature algorithm demonstrated in the feature database matching process is what causes the difference in the horizontal coordinates of experiments. Some feature algorithms, such as KAZE and AKAZE, are stable in extraction and always match, and the number of feature classes with a high number of matches is higher. However, when ORB and FREAK use binary descriptor matching, which is less stable than the gradient descriptor, the number of feature classes with a high number of matches is lower, and its maximum  $FMN$  value is smaller.

Figure 4 shows the correct feature matching rate of each image in the test set  $C$  and the feature databases based on different feature algorithms. From the figure, it can be seen that as  $FMN$  gradually increases, the feature correct matching rate of the test image basically shows an increasing trend, and the increasing trend gradually becomes slower. This indicates that features with more matches in the feature database are more stable in the image of that region, and filter features, which always appear in that experimental region, can always match with the features extracted from the test images.



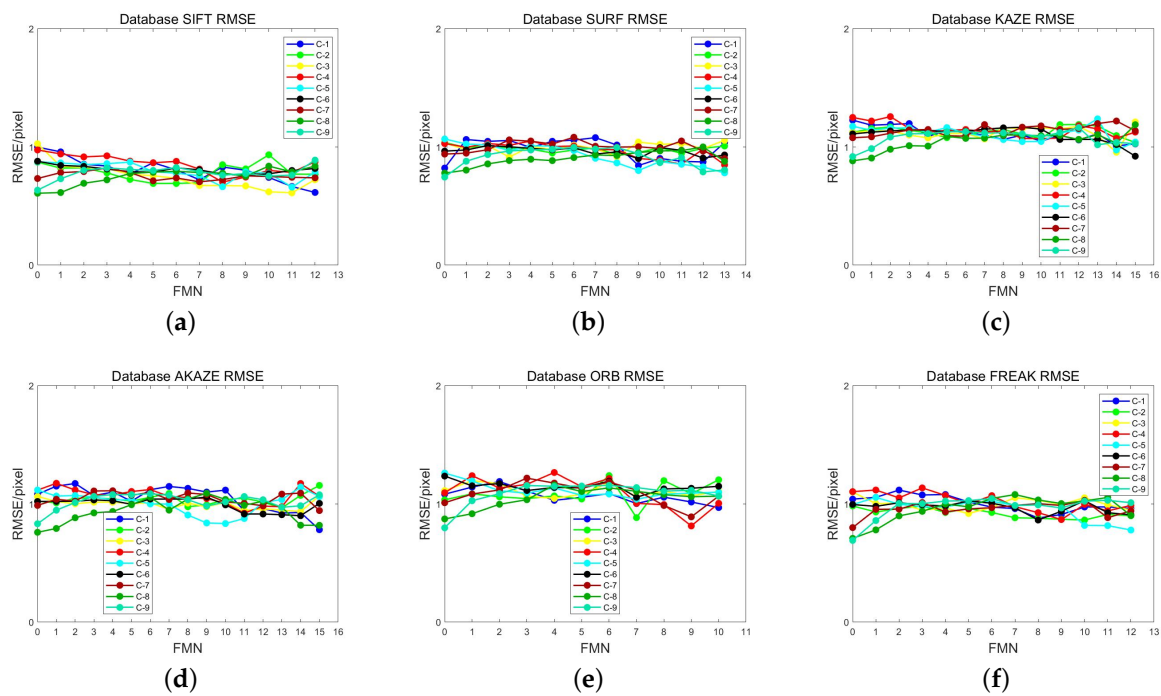


**Figure 4.** CMR of different *FMN* based on different feature database. (a) SIFT. (b) SURF. (c) KAZE. (d) AKAZE. (e) ORB. (f) FREAK.

Although the larger the *FMN*, the more stable the feature classes extracted from the feature database, it is not true that the higher the value of *FMN* the better the test image matches the feature database. It can also be observed from Figure 4 that the growth rate of *CMR* for matching the feature database to the test images is not obvious after *FMN* = 6. Especially for the AKAZE feature database, there is a significant decrease in *CMR* in the matching of test images C-1, C-6, because the larger the *FMN*, the smaller the number of extracted feature classes. The limited number of feature classes can also not reach a particularly high *CMR*. Therefore, in the comparison experiment with the reference image, we generally set *FMN* = 6, which has both a high *CMR* and a certain number of feature classes.

Figure 5 shows *RMSE* values after matching the test images with the feature databases based on different feature algorithms according to different *FMN* values. As can be seen from the figure, as *FMN* gradually increases, the *RMSE* obtained by the feature database matching methods based on different feature algorithms stays basically the same. As shown in Table 4, the *RMSE* fluctuations of the feature database matching results based on various feature operators are fewer than 0.5 pixels under various test images. This shows that *FMN* has little influence on the matching results and also little effect on the precision of geometric registration following a match.

Combined with the above description, it can be concluded that with the gradual increase of *FMN*, the overall *CMR* based on the stable feature database matching method shows an increasing trend, while *RMSE* is not affected by the change in the number of filtering matches. This indicates the feasibility and effectiveness of filtering stable feature matching strategies based on the stable feature database matching method. Moreover, when using the feature database matching method to correct remote sensing images, we can typically set *FMN* = 6 to filter feature classes for matching to ensure that the number of features extracted from the feature database after filtering according to *FMN* is sufficient and the number of correct matches is certain.



**Figure 5.** RMSE of different FMN based on different feature databases. (a) SIFT. (b) SURF. (c) KAZE. (d) AKAZE. (e) ORB. (f) FREAK.

**Table 4.** RMSE (pixel) fluctuations for different feature database matching.

Algorithm	C-1	C-2	C-3	C-4	C-5	C-6	C-7	C-8	C-9
SIFT	0.3835	0.2424	0.4150	0.2715	0.2144	0.1203	0.1174	0.2299	0.2544
SURF	0.2573	0.1032	0.1405	0.2023	0.2872	0.1100	0.1811	0.2214	0.2519
KAZE	0.2178	0.1520	0.2553	0.1853	0.2635	0.2448	0.1415	0.3060	0.2463
AKAZE	0.3909	0.1901	0.1335	0.2526	0.3090	0.1574	0.1673	0.3305	0.2536
ORB	0.2203	0.3554	0.1493	0.4519	0.2214	0.1793	0.3277	0.2868	0.3591
FREAK	0.2366	0.1192	0.1809	0.2683	0.2764	0.1639	0.2306	0.3709	0.3540

### 3.3.2. Comparison Experiment between Direct Matching and Feature Database Matching

This section contrasts the feature database matching method with the direct application of the reference image matching method to evaluate the proposed method. The comparison evaluation parameters are *CMR*, *RMSE*, and *Time*. *FMN* = 6 is set by the explanation in the previous section.

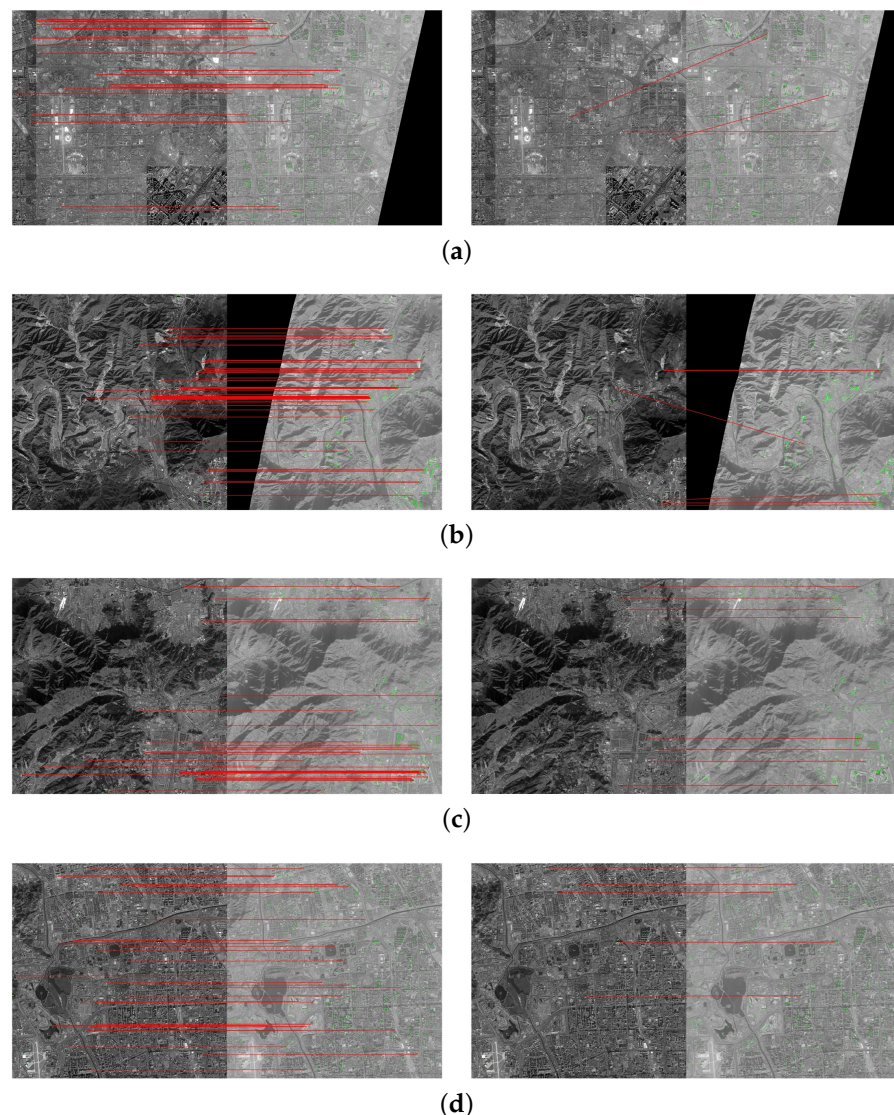
The trained feature database in Section 3.3.1 contains all feature classes with matching numbers. In order to further reduce the storage space and improve the matching efficiency, all feature classes with matching numbers fewer than 6 in the feature database can be deleted. The experiments in this section are conducted on the basis of the deleted feature database. Furthermore, in the block experiment, the block is not directly involved in matching when the number of feature classes obtained from the feature database in the test image block latitude and longitude range is less than 10, which allows for reducing the time required for matching.

Only the reference image *A* and the feature database were used for comparison in the experiments, and the training image was not used for reference image experiments. This is because the longitude and latitude ranges of the test image and the single training image may overlap, but not necessarily the with exact same scene. However, the feature database contains the range within the whole reference region, and the features within the test image region can be filtered according to the latitude and longitude. If the search ranges of the two methods are not the same for training image matching and feature database matching,



they cannot effectively illustrate the effectiveness of the feature database method. Finally, a reference image *A* is intercepted based on the latitude and longitude of the test image to create different reference image slices. These reference image slices are used for feature matching with the test images and compared with the feature database.

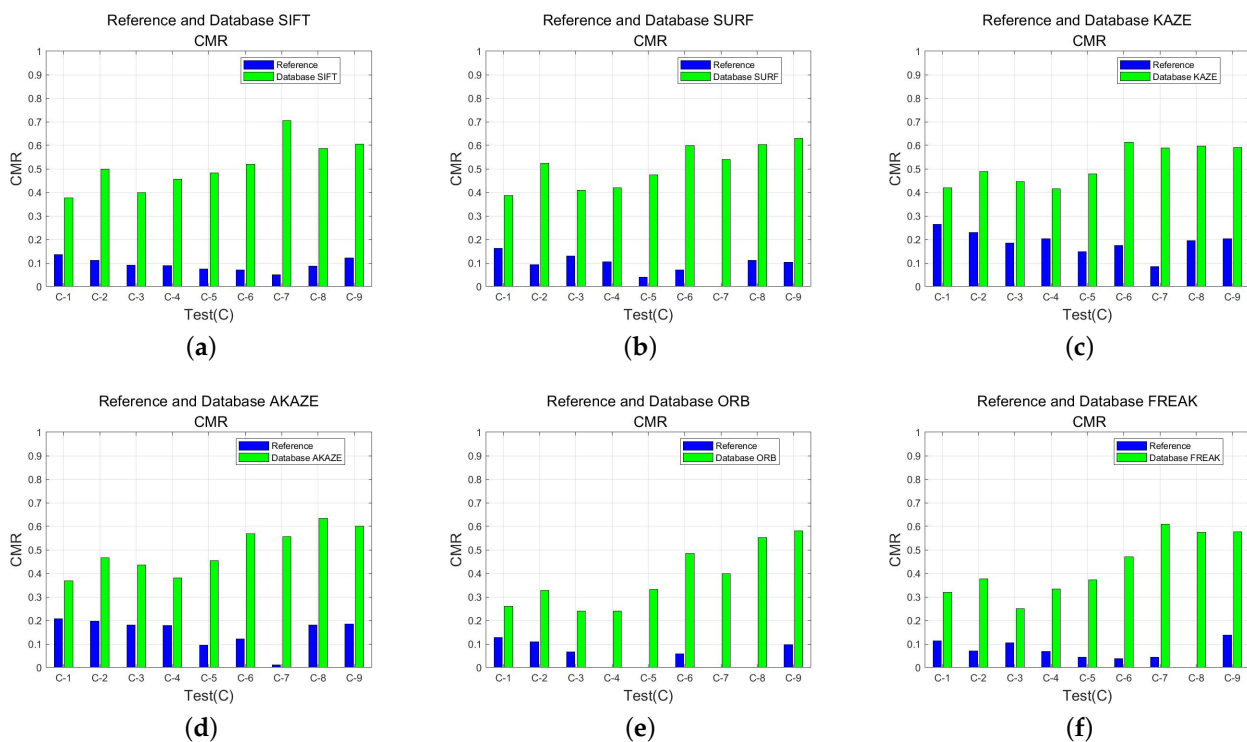
Figure 6 shows that the test image C-5 matches with the ORB feature database and the reference image, respectively. The left side images of Figure 6a–d are matched with the feature database, and the features from the feature database are displayed on the reference image for a more visual comparison with the right side images. The image on the right side matches the reference image with ORB. Based on the same number of extracted features, it can be seen that the feature database matching method obtains more pairs of matching points and the correspondence of the feature points is more accurate. With direct matching with the reference image using ORB features, the correspondence of the features may be inaccurate, as shown in Figure 6a,b. Although the right side image of Figure 6c,d shows accurate point correspondence, its number of matched point pairs is small, while the feature database matching method obtains a larger number of matched point pairs and is more stable than the right side.



**Figure 6.** Comparison of the reference image matching method and the feature database matching method based on C-5. (a–d) C-5 image block matching. The left image is the result of matching with the feature database, and the right image is the result of matching with the reference image. The left end of each image connecting line is the reference image, and the right end is the test image.

The comparison of the *CMR* between the direct matching method with the reference image and the feature database matching methods based on different feature algorithms is shown in Figure 7. In the experiment, if the test image blocks and the reference image slice blocks or feature database matching point pairs exceed 10, the matching is considered stable, and anything less than this threshold is considered unstable. It is observed that there are cases where *CMR* is 0 in the figure, such as C-7 in Figure 7b, C-5 in Figure 7e, etc., because in the reference image slices and the test image in the block matching process, the number of matches obtained by none of the blocks exceeds the set threshold, and we consider the matching to fail at this time.

It is obvious that directly matching with the reference image using classical operators does not match all test images successfully, but by using the simple stable feature database to match, every test image in the test set C can be correctly matched. This is because our proposed feature database matching method, which filters out stable feature classes and assembles descriptors of stable features under multiple observation conditions, has a higher possibility of successful matching with different optical satellite images. Therefore, in the comparison experiments, the feature database can achieve stable matching with all test images in the test set. Furthermore, in the case of both methods, the feature database matching method based on different feature operators has essentially an over 30% greater correct feature point matching rate than direct matching using this feature operator, as shown in Table 5.



**Figure 7.** *CMR* comparison of the direct matching method and the feature database matching method. (a) SIFT. (b) SURF. (c) KAZE. (d) AKAZE. (e) ORB. (f) FREAK.

Figure 8 shows the *RMSE* comparison between the direct matching methods and feature database matching methods based on different feature algorithms. The *RMSE* of the feature database matching approach based on different feature operators is approximately equivalent to that of the direct matching with the reference image, with an absolute difference average value of fewer than 0.3 pixels. The *RMSE* of the matching instability case will not be shown in the graph. As can be seen, geometric registration on large-scale remote sensing images can be performed with a certain degree of accuracy using the feature database matching method. For more details, see Table 5.

Figure 9 compares the matching time for direct matching and feature database matching methods based on different feature algorithms. The figure clearly shows that the feature database matching method always takes less time than the direct matching method using the reference image slice, because the feature database matching method eliminates the step of extracting features from the reference image. Table 5 shows that, compared to direct matching, the feature database matching method based on various feature operators typically reduces the time taken by more than 40%.

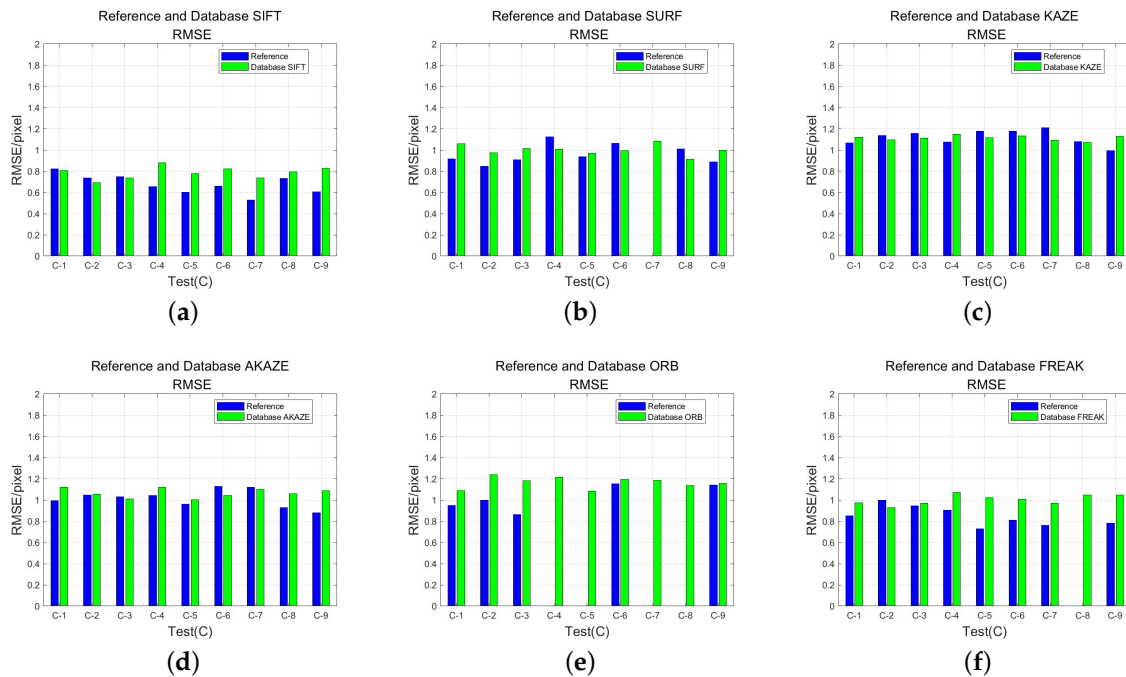


Figure 8. RMSE comparison of direct matching method and feature database matching method. (a) SIFT. (b) SURF. (c) KAZE. (d) AKAZE. (e) ORB. (f) FREAK.

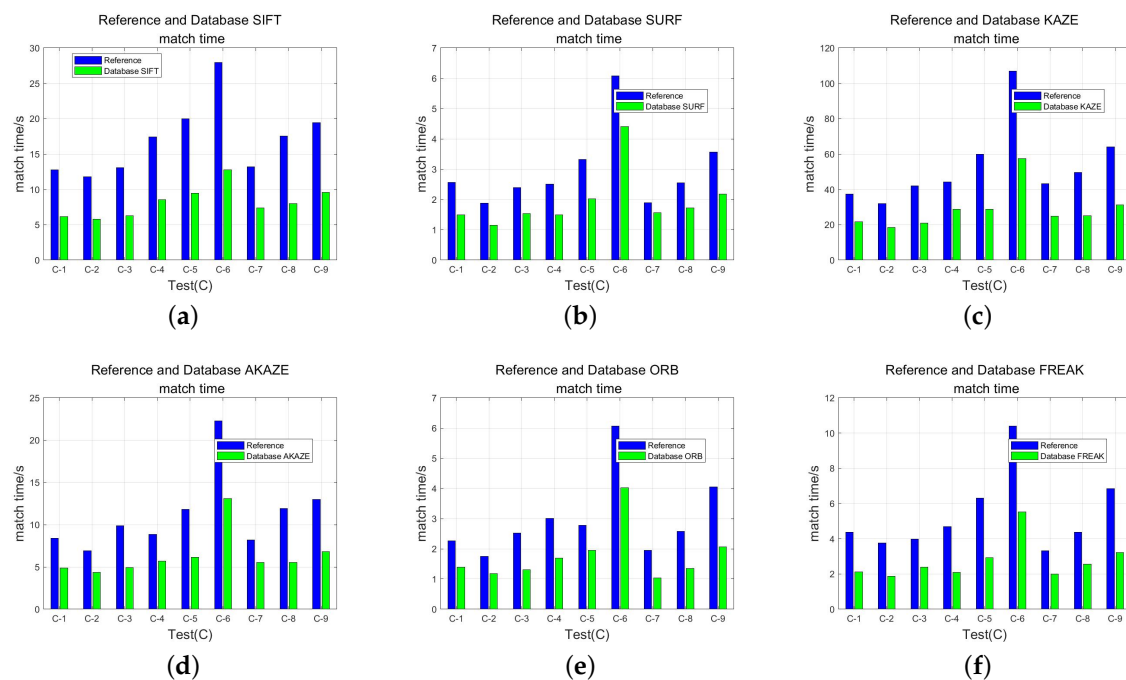


Figure 9. Match time comparison of direct matching method and feature database matching method. (a) SIFT. (b) SURF. (c) KAZE. (d) AKAZE. (e) ORB. (f) FREAK.

**Table 5.** Comparison of parameters between direct matching method and feature database matching method.

Algorithm	Stable Direct Match/Image	Stable Feature Database Matching/Image	Average CMR Increase/%	RMSE Absolute Difference Mean/pixel	Average Time Reduction/%
SIFT	9	9	42.23	0.1251	51.31
SURF	8	9	40.38	0.101	36.83
KAZE	9	9	32.78	0.064	45.66
AKAZE	9	9	34.54	0.0803	43.08
ORB	5	9	28.76	0.1508	40.33
FREAK	8	9	33.61	0.1685	48.69

### 3.3.3. Difference between Feature Database

Six classical feature algorithms—SIFT, SURF, KAZE, AKAZE, ORB, and FREAK—are selected to construct simple and stable feature databases in the above experiments. The feature databases constructed by each operator perform well in matching comparison experiments, and it can be seen that the feature database matching algorithm is universal for a variety of feature algorithms. This subsection performs matching experiments based on test image C-9 to compare the performance of feature databases based on different feature algorithms.

Figure 10 compares the results of the test image C-9 matching with the feature database based on different feature methods. From Figure 10a, it is clear that the matching rate increases gradually as the value of *FMN* increases and that the upward trend progressively slows down. The upward trend of the correct matching rate of each feature database in this figure is roughly the same. At an  $FMN \geq 9$ , the *CMR* of the test image that matches the ORB feature database features decreases, probably due to the insufficient number of stable filtered feature classes.

From Figure 10b,c, it can be seen that the number of feature classes extracted from the feature databases and the number of successful matches with the test image gradually decrease with the gradual increase of *FMN*, and the trend is gradually slowing. At different *FMN* values, the number of stable feature classes filtered from the KAZE, AKAZE, SIFT, and SURF feature databases is relatively high and the number of feature classes obtained from the ORB and FREAK feature databases is low. The totality of the “blob” detector is generally better than the “corner” point detector [35], so the “blob” detector can retain more stable invariant feature classes in the iterative matching process of the feature database.

From Figure 10d, we can see that the KAZE operator takes the longest time among the feature database matching methods. Due to the complexity of the KAZE operator, it takes a long time even if the matching time is approximately half as long using the feature database method matching. It is observed that all curves in the figure show a slight downward trend because the number of filtered feature classes gradually decreases with increasing *FMN* values and the time to read the features from the feature database also decreases. Furthermore, in the blocking experiment, when the number of feature classes obtained from the feature database in the test image blocking latitude and longitude range is less than 10, the block is not directly involved in matching, which also causes a decrease in time.

In combination with the above analysis, the performance comparison between feature databases is mainly determined by the use of feature algorithms within the feature database. In the proposed feature database matching method, KAZE is relatively stable, but it takes a long time. When compared to KAZE, AKAZE requires much less time and has fewer stable features, but the construction of the nonlinear scale space still takes a certain amount of time. SIFT and SURF are not as stable as KAZE and AKAZE and are not as fast as ORB and FREAK. ORB and FREAK are the most efficient, but the combination of corner detectors and binary descriptors is not as stable as blob detectors and floating-point descriptors. We can construct simple and stable feature databases according to different application requirements. If the tendency is for fast matching and high matching efficiency, automatic



implementation algorithms such as ORB or FREAK may be selected for the construction of feature databases. If the tendency is to have good matching stability and a large number of stable feature classes, the feature database can be constructed by choosing algorithms such as KAZE or AKAZE.

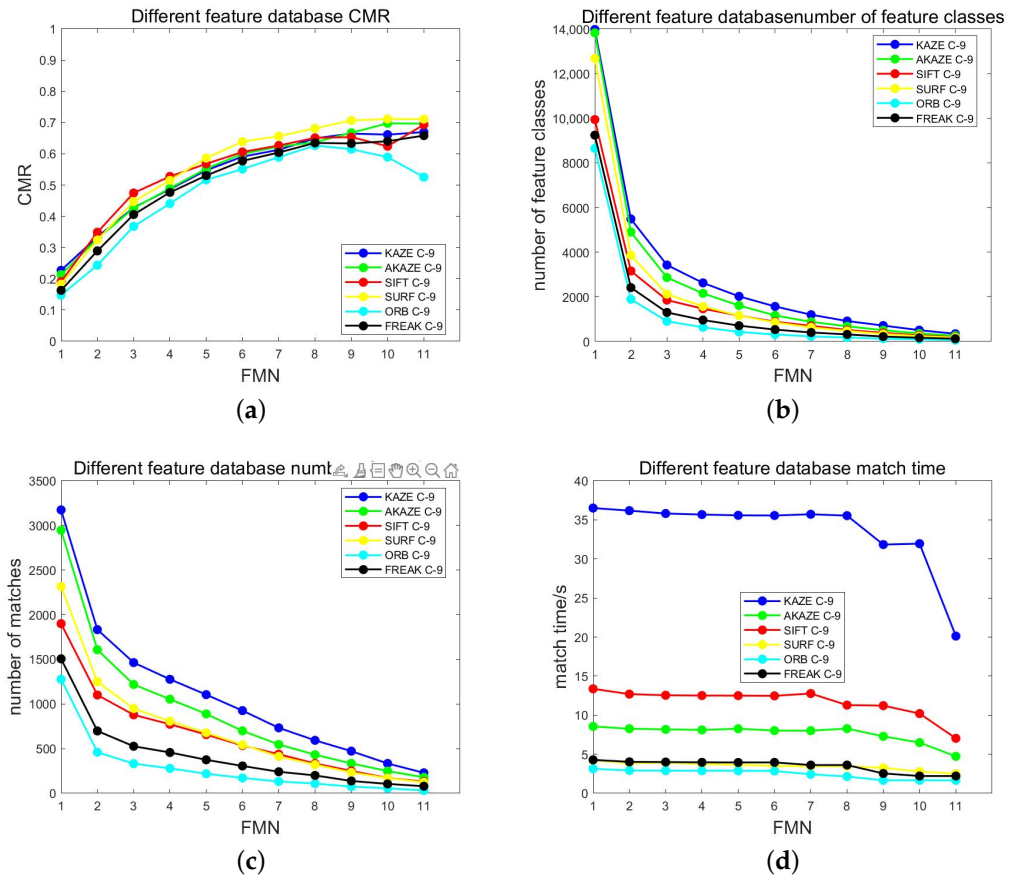


Figure 10. Different feature database comparison (a) CMR. (b) the number of feature classes. (c) the number of matches. (d) match time.

#### 4. Discussion

The experiments in Section 3 demonstrate the usefulness and stability of the feature database matching approach using data set *D* and different feature extraction methods.

The feature database matching method based on different feature algorithms can achieve 100% stable matching of the test image on the test set *C*, but the direct matching of the test image with the reference image may not match successfully. When both matching methods are successful, the feature database matching approach has a higher rate (mostly above 30%) of correctly matching feature points than direct matching. This is because the simple and stable feature database matching method takes into account the differences in feature descriptors under multiple observation conditions. The proposed method uses an iterative matching method to train stable feature points and aggregate multiple descriptors at the same feature point location, which greatly increases the probability of successful matching with the test image. Moreover, the results from the feature database matching approach are not significant deviations from the direct matching results in terms of *RMSE*. Additionally, compared with direct matching, the feature database matching method saves a large amount of time because it skips the extraction of features from the reference image. For most feature algorithms, the matching of the feature database has been shown to take less time than direct matching by more than 40%.

However, there are still some parts to be improved in this paper. First, according to the iterative matching strategy of the feature database in this paper, it is theoretically possible to automatically update the feature database while continuously inputting the test

image. However, training and testing are separated in the paper, and there is no mention that the test image can also update the feature database. This is because the lack of coding ability makes feature clustering take some time, and the matching process plus the feature database update process will become less efficient. Second, the time to extract the features from the feature database is almost negligible compared with the time to extract the features from the reference image. However, the experimental results show that inherently efficient operators such as SURF do not save as much time as other operators using the feature database method; because the descriptors of SURF are floating points, it takes more time to read them by the program written by itself. This can be solved by improving the database reading procedure. Third, the current experimental method has been verified only for optical remote sensing images, while this framework is theoretically applicable to a wide range of sensors.

In the future, we hope to optimize the program so that we can also train based on the test image when the test image is matched and perform fast updates to the feature database. Furthermore, we will build the feature database using more stable feature algorithms such as histogram of orientated phase congruency (HOPC) [36] and radiation-variation insensitive feature transform (RIFT) [37], among others, that are suitable for multi-modal image matching to experiment on multi-modal images. Based on the iterative approach of the feature database, the invariant features within the scene are found by the input multi-modal images, and the descriptors of multiple styles under the same location are clustered to make the feature database approach more universal.

## 5. Conclusions

This paper proposes a fast-matching method for optical remote sensing images based on stable and simple feature databases. The advantages of the simple and stable feature database matching method can be summarized as follows.

1. **Simplicity:** This feature database matching method stores the features extracted from the reference image in the feature database simply and effectively for subsequent matching. Since features do not need to be extracted from the reference data each time matching is performed, this reduces the amount of storage space required for the reference data and speeds up geometric correction and remote sensing image matching.
2. **Stability:** The feature database extracts features from images of the same region with different time phases and trains stable invariant features (constructs invariant feature point sets) by iterative matching. It increases the correct matching rate by extracting stable feature classes with numerous matches, being adaptable to remote sensing image matching under multiple observation conditions.
3. **Scalability:** The method in this paper has good reconfigurability because, in addition to the feature algorithms mentioned in the paper, other feature algorithms can be modified by the image resolution of remote sensing images, various geomorphological features of the target area, and various sensor types for acquiring images. The feature database matching technique can be made even more effective in the future by including faster feature algorithms.

Instead of repeatedly extracting feature points from the reference image, the fast matching method based on a simple stable feature database can select existing feature points in the corresponding area of the image in the feature database, potentially reducing the storage space of the reference data and improving the efficiency of image processing. Additionally, stable invariant features can be extracted to take part in matching by filtering the stable invariant feature matching parameters to increase the matching accuracy. This matching technique is highly flexible and can be used with many different feature extraction algorithms.



**Author Contributions:** Conceptualization, Z.Z. and H.L.; methodology, Z.Z. and H.L.; software, Z.Z. and H.L.; validation, Z.Z.; formal analysis, Z.Z.; investigation, Z.Z.; resources, Z.Z. and H.L.; data curation, Z.Z.; writing—original draft preparation, Z.Z.; writing—review and editing, Z.Z., H.L. and H.Y.; visualization, Z.Z.; supervision, Z.Z.; project administration, H.L. and H.Y.; funding acquisition, H.L. and H.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Key Research Program of Frontier Sciences, Chinese Academy of Sciences, under Grant ZDBS-LY-JSC036.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** We thank the reviewers for their valuable comments and suggestions. We also would like to thank the production team for revising the format of the manuscript.

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

## References

1. Yue, Z.; Fan, D.; Dong, Y.; Ji, S.; Li, D. A generation method of spaceborne lightweight and fast matching. *J. Geo-Inf. Sci.* **2022**, *24*, 925–939.
2. Zhou, G.; Zhang, R.; Liu, N.; Huang, J.; Zhou, X. On-Board Ortho-Rectification for Images Based on an FPGA. *Remote Sens.* **2017**, *9*, 874. [[CrossRef](#)]
3. Chen, B.; Huang, B.; Xu, B. Comparison of Spatiotemporal Fusion Models: A Review. *Remote Sens.* **2015**, *7*, 1798–1835. [[CrossRef](#)]
4. Shen, H.; Meng, X.; Zhang, L. An Integrated Framework for the Spatio-Temporal-Spectral Fusion of Remote Sensing Images. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 7135–7148. [[CrossRef](#)]
5. Thomas, C.; Ranchin, T.; Wald, L.; Chanussot, J. Synthesis of Multispectral Images to High Spatial Resolution: A Critical Review of Fusion Methods Based on Remote Sensing Physics. *IEEE Trans. Geosci. Remote Sens.* **2008**, *46*, 1301–1312. [[CrossRef](#)]
6. Ma, Y.; Li, H.; Gu, H. A Study of Fast Change Detection Algorithm Based on Feature Library of Remote Sensing Imagery. In Proceedings of the 2011 International Symposium on Image and Data Fusion, Tengchong, China, 9–11 August 2011; pp. 1–3. [[CrossRef](#)]
7. Zhang, C.; Feng, Y.; Hu, L.; Tapete, D.; Pan, L.; Liang, Z.; Cigna, F.; Yue, P. A Domain Adaptation Neural Network for Change Detection with Heterogeneous Optical and SAR Remote Sensing Images. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *109*, 102769. [[CrossRef](#)]
8. Zhong, Y.; Liu, W.; Zhao, J.; Zhang, L. Change Detection Based on Pulse-Coupled Neural Networks and the NMI Feature for High Spatial Resolution Remote Sensing Imagery. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 537–541. [[CrossRef](#)]
9. Chen, Z.; Chi, Z.; Zinglensen, K.B.; Tian, Y.; Wang, K.; Hui, F.; Cheng, X. A New Image Mosaic of Greenland Using Landsat-8 OLI Images. *Sci. Bull.* **2020**, *65*, 522–524. [[CrossRef](#)]
10. Jiang, Y.; Xu, K.; Zhao, R.; Zhang, G.; Cheng, K.; Zhou, P. Stitching Images of Dual-Cameras Onboard Satellite. *ISPRS J. Photogramm. Remote Sens.* **2017**, *128*, 274–286. [[CrossRef](#)]
11. Li, X.; Hui, N.; Shen, H.; Fu, Y.; Zhang, L. A Robust Mosaicking Procedure for High Spatial Resolution Remote Sensing Images. *ISPRS J. Photogramm. Remote Sens.* **2015**, *109*, 108–125. [[CrossRef](#)]
12. Chen, Q.H.; Liu, X.G.; Gao, W.; Liu, T.L. An Automatic Ground Control Point Matching Based on GCP Chip Database for Remote Sensing Images. In Proceedings of the 2009 International Conference on Image Analysis and Signal Processing, Linhai, China, 11–12 April 2009; pp. 13–17. [[CrossRef](#)]
13. Tang, P.; Zheng, K.; Shan, X.; Hu, C.; Huo, L.; Zhao, L.; Li, H. Framework of remote sensing image automatic processing with “invariant feature point set” as control data set. *J. Remote Sens.* **2016**, *20*, 1126–1137.
14. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [[CrossRef](#)]
15. Ke, Y.; Sukthankar, R. PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 27 June–2 July 2004; Volume 2, p. II. [[CrossRef](#)]
16. Mikolajczyk, K.; Schmid, C. A Performance Evaluation of Local Descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 1615–1630. [[CrossRef](#)]
17. Bay, H.; Tuytelaars, T.; Van Gool, L. SURF: Speeded Up Robust Features. In Proceedings of the Computer Vision—ECCV, Graz, Austria, 7–13 May 2006; Lecture Notes in Computer Science; Leonardis, A., Bischof, H., Pinz, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 404–417. [[CrossRef](#)]
18. Sedaghat, A.; Mokhtarzade, M.; Ebadi, H. Uniform Robust Scale-Invariant Feature Matching for Optical Remote Sensing Images. *IEEE Trans. Geosci. Remote Sens.* **2011**, *49*, 4516–4527. [[CrossRef](#)]

19. Alcantarilla, P.F.; Bartoli, A.; Davison, A.J. KAZE Features. In Proceedings of the Computer Vision—ECCV, Florence, Italy, 7–13 October 2012; Lecture Notes in Computer Science; Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 214–227. [\[CrossRef\]](#)
20. Alcantarilla, P.; Nuevo, J.; Bartoli, A. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In Proceedings of the British Machine Vision Conference, Bristol, UK, 9–13 September 2013; British Machine Vision Association: Bristol, UK, 2013; pp. 13.1–13.11. [\[CrossRef\]](#)
21. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An Efficient Alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 October 2011; pp. 2564–2571. [\[CrossRef\]](#)
22. Leutenegger, S.; Chli, M.; Siegwart, R.Y. BRISK: Binary Robust Invariant Scalable Keypoints. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 October 2011; pp. 2548–2555. [\[CrossRef\]](#)
23. Alahi, A.; Ortiz, R.; Vanderghyest, P. FREAK: Fast Retina Keypoint. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 510–517. [\[CrossRef\]](#)
24. Ma, J.; Jiang, J.; Zhou, H.; Zhao, J.; Guo, X. Guided Locality Preserving Feature Matching for Remote Sensing Image Registration. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 4435–4447. [\[CrossRef\]](#)
25. Li, Z.; Yue, J.; Fang, L. Adaptive Regional Multiple Features for Large-Scale High-Resolution Remote Sensing Image Registration. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–13. [\[CrossRef\]](#)
26. Chen, S.; Zhong, S.; Xue, B.; Li, X.; Zhao, L.; Chang, C.I. Iterative Scale-Invariant Feature Transform for Remote Sensing Image Registration. *IEEE Trans. Geosci. Remote Sens.* **2021**, *59*, 3244–3265. [\[CrossRef\]](#)
27. Kelman, A.; Sofka, M.; Stewart, C.V. Keypoint Descriptors for Matching Across Multiple Image Modalities and Non-linear Intensity Variations. In Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 17–22 June 2007; pp. 1–7. [\[CrossRef\]](#)
28. Martins, P.; Carvalho, P.; Gatta, C. On the Completeness of Feature-Driven Maximally Stable Extremal Regions. *Pattern Recognit. Lett.* **2016**, *74*, 9–16. [\[CrossRef\]](#)
29. Ma, W.; Wu, Y.; Liu, S.; Su, Q.; Zhong, Y. Remote Sensing Image Registration Based on Phase Congruency Feature Detection and Spatial Constraint Matching. *IEEE Access* **2018**, *6*, 77554–77567. [\[CrossRef\]](#)
30. Agrawal, M.; Konolige, K.; Blas, M.R. CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching. In *Computer Vision—ECCV 2008*; Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5305, pp. 102–115. [\[CrossRef\]](#)
31. Rosten, E.; Drummond, T. Machine Learning for High-Speed Corner Detection. In Proceedings of the Computer Vision—ECCV, Graz, Austria, 7–13 May 2006; Lecture Notes in Computer Science; Leonardis, A., Bischof, H., Pinz, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 430–443. [\[CrossRef\]](#)
32. Calonder, M.; Lepetit, V.; Strecha, C.; Fua, P. BRIEF: Binary Robust Independent Elementary Features. In Proceedings of the Computer Vision—ECCV, Crete, Greece, 5–11 September 2010; Lecture Notes in Computer Science; Daniilidis, K., Maragos, P., Paragios, N., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 778–792. [\[CrossRef\]](#)
33. Wu, Y.; Ma, W.; Gong, M.; Su, L.; Jiao, L. A Novel Point-Matching Algorithm Based on Fast Sample Consensus for Image Registration. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 43–47. [\[CrossRef\]](#)
34. Xiang, Y.; Wang, F.; You, H. OS-SIFT: A Robust SIFT-Like Algorithm for High-Resolution Optical-to-SAR Image Registration in Suburban Areas. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 3078–3090. [\[CrossRef\]](#)
35. Moghimi, A.; Celik, T.; Mohammadzadeh, A.; Kusetogullari, H. Comparison of Keypoint Detectors and Descriptors for Relative Radiometric Normalization of Bitemporal Remote Sensing Images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 4063–4073. [\[CrossRef\]](#)
36. Ye, Y.; Shan, J.; Bruzzone, L.; Shen, L. Robust Registration of Multimodal Remote Sensing Images Based on Structural Similarity. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 2941–2958. [\[CrossRef\]](#)
37. Li, J.; Hu, Q.; Ai, M. RIFT: Multi-Modal Image Matching Based on Radiation-Variation Insensitive Feature Transform. *IEEE Trans. Image Process.* **2020**, *29*, 3296–3310. [\[CrossRef\]](#)

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.