# Fine-grained localisation using feature matching

*

1st Xinyu Zeng
*School of Engineering and IT*
*The University of Melbourne*
Melbourne, Australia
zexz1@student.unimelb.edu.au

2th Zhiyi Hu
*School of Engineering and IT*
*The University of Melbourne*
Melbourne, Australia
zhiyih1@student.unimelb.edu.au

***Abstract*—This report researches a small indoor/outdoor fine-grained geolocation problem. Experiments are designed to compare different Hash to improve the efficiency of image matching and using SIFT algorithm to exclude key points of images irrelevant to matching. It also included error analysis and future improvement.**

***Index Terms*—Computer Vision, Fine-grained, Feature Matching, Hash, SIFT, BF Matcher, FLANN-Based Matcher**

## I. Introduction

In traditional computer vision research, the target objects of image analysis are generally classified by object categories, however, in practical applications, image objects often come from different subclass categories at a finer granularity level under a traditional category. Fine-grained level image analysis is a popular research topic in the field of computer vision, and its goal is to study visual analysis tasks such as localization, recognition and retrieval of object subclasses in fine-grained level images, which has wide application value in real scenarios. This report explores the problem of small indoor/outdoor fine-grained geolocation image analysis based on "similarity" feature matching to identify the location of captured images. By designing experiments based on traditional hash algorithms, this report proposes an approximate nearest neighbor matching algorithm based on SIFT descriptors for image similarity comparison between the training sets and test sets to infer the location data of the test set.

## II. Related work

After reading the related literature, we decided to use the Hash algorithm to compare image similarity and SIFT to match key points of images. Hashing method is used for large no of pixels it can produce great results with very high accuracy and preparing perceptual integrating other features like texture, shape will provide best results. [1] SIFT is a computer vision algorithm used to detect and characterize local features in images, which finds extreme points in spatial scales and extracts their position, scale, and rotation invariants. [2]

The whole experimental process is designed as the detection process of similar images. We assume that the matched training set and the test set images have the same data location. Firstly, each image in the training set and test set is encoded or feature extracted in the same way (generally in the form of a feature vector), and then the distance between that encoding or that feature vector and the encoding or vector of the image in the database is calculated as the similarity between images, and the similarity is ranked, and the images with the top similarity or matching the requirement are displayed.

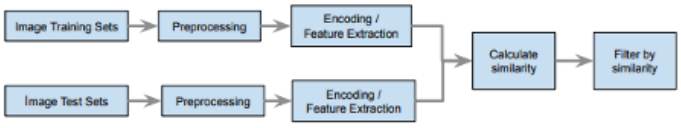


Fig. 1. Project Design for Experiment

## III. DataSet

The dataset used in the report is a collection of images taken in and around the Museum of Art (Getty Center, Los Angeles, U.S.A.) and then rendered by Google Street View images, simulating a camera with a 73.7 degree horizontal multiple 53.1 degrees vertical of view. The optical center of the camera is in the center of the lens and there is no radial distortion in the lens. The whole dataset has been split into two categories, 7500 training images and 1200 test images. All the images are named with a unique id number. Only the training dataset contains location coordinates (x,y) values derived from a mapping algorithm and can be assumed that the image's position is accurately reflected in the real world.

## IV. Methodology

### A. Hashing Algorithm

In the engineering field, the most commonly used image similarity algorithm is hash, which are average hash algorithm (aHash), perceptual hash algorithm(pHash) and difference hash algorithm (dHash). These three methods of the Hash algorithm is to get the hash value of the image, and then compare the Hamming distance of two images hash value to measure whether the two images are similar. If two images are more similar, the smaller the Hamming distance of their hash numbers. [3]

*a) Average hash algorithm(aHash):* By compressing the original image (8×8) and grayscale processing, the mean value of the compressed image pixels is calculated, and the 64 pixels of the 88 images are compared with the mean value, and the value greater than the mean value is 1 and less than the mean value is 0. The resulting 64-bit binary code is the aHash value of the original image.

*b) Difference hash algorithm (dHash):* Compared with aHash, dHash also performs image compression and grayscale processing, and then generates fingerprint information by comparing the previous pixel in each row with the next pixel, where greater than is 1 and less than is 0.

*c) Perceptual hash algorithm (pHash):* By compressing the original image (32×32) and grayscale processing, and use DCT(discrete cosine transform) converts intensity data into frequency data and produces a new 8×8 block of low frequency value data. Compute the average DCT value and set the hash bits to zero or one based on the mean value. The 64-bit fingerprint information is obtained as the phash value of the original image.

Hash is a common method of image similarity analysis. Choosing it to filter images can help us eliminate some images that are not very similar to images, reduce subsequent processing pressure and improve efficiency.

### B. Similarity measurement

We used Hamming distance to measure the similarity for two hashes. The normalized Hamming distance (HD) [4] between pair of image hases is given by the following equation:

$$HD = \frac{\sum_{n=1}^{L} H_i(n) - H_m(n)))}{L} \tag{1}$$

*Where L=the length of the hash value, Hi=the hash value of the i th and Hm=the hash value of the m th.*

### C. SIFT

The SIFT algorithm was proposed by Professor David Lowe [2] in 1999. SIFT mainly selects potential keypoints by eliminating keypoints with low contrast and high edge response, and then calculates the keypoint descriptors needed to match the features.[2] The SIFT algorithm is designed with affine invariance, viewpoint invariance, rotation invariance and illumination invariance.

*a) Brute-Force Matcher(BF Matcher):* BF Matcher will select a descriptor of the feature in the first group and balance it with all possible alternatives in the second group by using some distance calculation. and returning the closest one. [5]

*b) FLANN-Based Matcher:* FLANN is a collection of optimization algorithms for nearest neighbor search on large data sets and high-dimensional features. The FLANN matcher is usually used together with the SIFT algorithm to search for matching image similarities. The matcher first tries to find the first two best matches for the key points. The ratio of the first closest distance to the second closest distance is then calculated to determine if the found points are correct. If the ratio is less than the threshold, then we consider them as good points; otherwise, these points will be rejected. The matches of these good points are plotted and the similarity score is calculated with the help of these good points. [6]

$$\begin{aligned} Similarityscore = (Lengthofgoodpoints \\ /numberofkeypoints) \times 100 \end{aligned} \tag{2}$$

SIFT is a commonly used image matching algorithm with a wide range of applications. It has good uniqueness and rich information, and is suitable for fast and accurate matching in massive feature databases. Even a few objects can generate a large number of SIFT feature vectors, and the scalability is also strong. BFMatch and FlannMatch are also commonly used feature matching algorithms, and we will choose the more suitable one based on practice.

### D. Experiment Process

The main idea of the experiment is to match the test image with the train image one by one, and find the predicted image that is most similar to the test image by using the "similarity" as a measurement standard, so that the coordinates of the training image are used as the coordinates of the test image.

The main process is to first exclude some training images that are significantly different from the test image through hash and Hamming distance, and then use SIFT to complete the extraction of feature points and feature descriptors of the training image and the test image. Then try to use the two algorithms of BFMatch and FlannMatch to match the extracted feature points, use to calculate the similarity between the test image and the training image, and get the training image with the greatest similarity to the test image. Then predict that the location of the test image is basically the same as the training image.

## V. Evaluation

In order to seek better experimental results, we conduct comparative research and optimization around the following aspects.

### A. BFMath vs FlannMath

Both BFMatch and FlannMath are common feature matching algorithms. We selected 500 training images as the test set to test the effects of the two algorithms. Operation efficiency and MAE are the main considerations.

| Macther Types | FlannBased Matcher | BFMatcher |
|---|---|---|
| Runing Time per img | 23s | 343s |
| Mean Absolute Error | 15.70533 | 13.40683 |

The above Table showed the processing speed of each image, and the effect of FlannMatch is also significantly better than that of BFMatch. This is because BFMatch will traverse all possible matching point pairs. And FlannMatch matches based on the nearest proximity. Thus from the perspective of accuracy, BFMatch is better than FlannMatch, because theoretically it can find the best matching result, but the difference between the two is not big in the current situation.

Comprehensive consideration, FlannMatch better meets our needs. We need to increase the operating speed within an acceptable error range.

### *B. Image Pre-Filter*

Because our data set is too large, if we simply compare each training image with the test image, it will take a lot of time. Hash and Hamming distance are a good way to filter images and improve operation. efficient. For the three common hash algorithms, dhash, phash and ahash, we mainly use operating efficiency as the evaluation criteria to determine which method is more suitable. We selected some images for testing and calculated their average running speed. The result is as follows in the following table:

| **Hash Types** | **aHash** | **pHash** | **dHash** | **No Hash** |
|---|---|---|---|---|
| **Returning time per img (HD>0.7)** | 58s | 343s | 23s | 1170s |

The table showed that after the use of the hash algorithm to filter the image, the processing speed of the image has been significantly accelerated, Thus the use of the hash algorithm can improve our overall efficiency. The results of the three hash algorithms can also be seen. Among them, dhash is the fastest, ahash is the second, and phash is the slowest. Because the phash algorithm is more complicated, the running speed is reduced while the accuracy is high. The speed of ahash is relatively fast, but the accuracy is not enough. And dhash is relatively good for both, in the case where we are more concerned about speed, dhash is undoubtedly a better choice.

### *C. The Threshold of Hamming Distance*

The choice of the threshold of the Hamming distance determines how many images will be considered for feature matching with the test image. If the threshold is set too high, if the hash algorithm is not stable, it may cause some training images that are more consistent with the test image to be filtered out , resulting in poor matching results for the test images.Therefore, we mainly tried the three options of 0.7, 0.75 and 0.8 for the threshold.Through testing, it is found that when the threshold value is 0.8, some images can get a good matching effect, but many images are prone to errors and cannot be matched to a suitable image.When the threshold is 0.75, the matching effect of most images is good, but for some outdoor images, their feature information is relatively similar, and there are trees as interference, 0.75 still cannot have a better result. When the threshold is 0.7, most outdoor images can also be matched successfully. Therefore, 0.7 is a more appropriate choice from the perspective of accuracy. The Fig.2 shows the different training images matched by an image under different threshold selections. The image with the most similar content is not obtained until the threshold is 0.7.

### *D. Set the Range of Keypoints*

Some images have many key points, (see Fig.3) especially some trees in outdoor images, which themselves contain a lot of key points, and these points may cause some useless

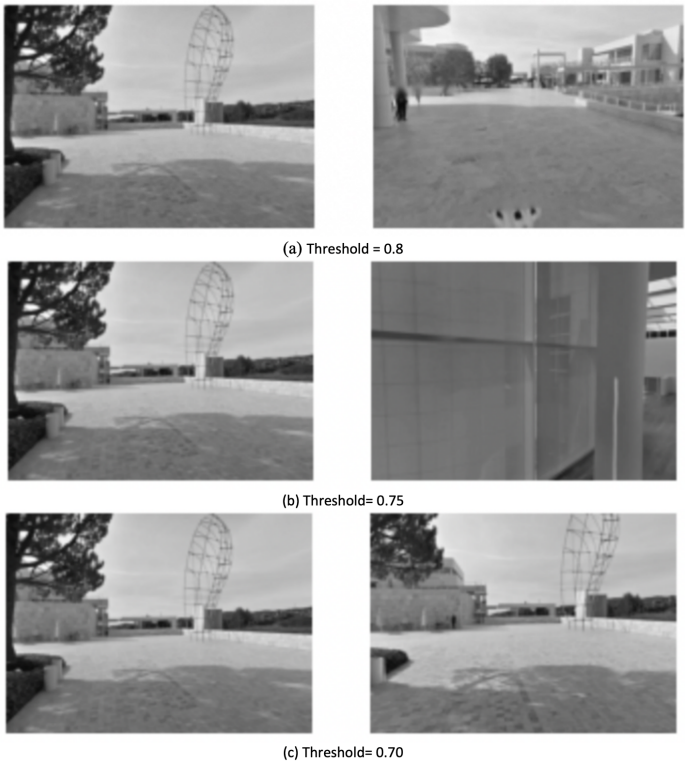


Fig. 2. Matching graphs caused by different thresholds

matching and affect the accuracy of matching. To this end, we set the number of key points to reduce the errors caused by trees and to obtain the correct matching results.

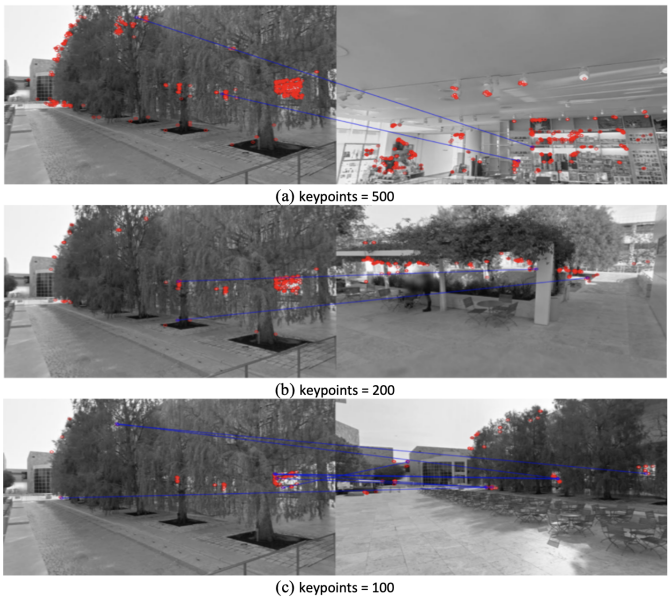


Fig. 3. Matching graphs caused by different keypoints

## VI. Improvement

Some aspects of the good performance of our model are that we can get a relatively satisfactory MAE value, and the accuracy of indoor images is relatively high. For some test images with more similar training images, we can also find closer images from it. The hash algorithm is also used to screen the images, which greatly improves the operating efficiency, and increases the accuracy of outdoor image matching by limiting the number of key points.

Through the analysis of the results, the poor performance of our model is mainly because some images can not find a matching image and the matched image is very different from the test image, which brings huge errors. Some walls in the test set are prone to fail to find a matching picture. (see Fig.4) When this happens, our approach is to set its coordinates to (0, 0),which is likely to bring great errors. Most of these walls are smooth, and it is difficult to find enough feature points to match after removing the color information. This is also a problem that needs to be resolved.

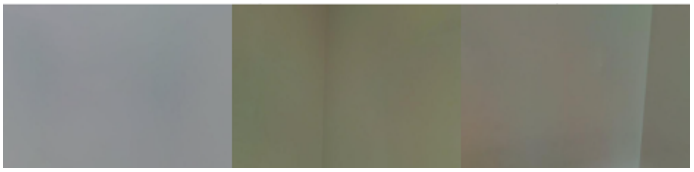

Fig. 4. Three colored Wall Image with different ID

The main causes of other large errors are: Outdoor images can easily not match accurate images, because most of the outdoor images contain some trees. Trees will bring a lot of key points, which may lead to some wrong matching and affect accuracy. Fig.5 is a failure example. Even though the key points are restricted, a good result is still not obtained.

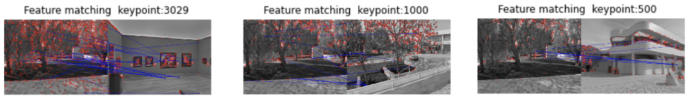


Fig. 5. Limitations of key point control

Another aspect that affects the low accuracy of outdoor image matching is that many outdoor buildings have similar appearances and similar materials.(see Fig.6) It is very likely that there are many similar features in the image but do not belong to the same location, resulting in relatively large errors.

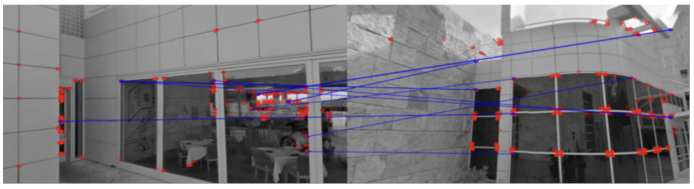

Fig. 6. Failed match due to the same material and style

There is also a very common and difficult to avoid error is that a certain local object in the image takes up a large proportion. When matching, the model will pay more attention to the characteristics of the local object, while ignoring the information of other parts. (see Fig.7) Some images with doors and corners as content are also difficult to accurately match in some cases, because their materials and structures are relatively similar, and even some picture frames on the wall are very similar, and it is difficult to accurately identify them after removing the color information.

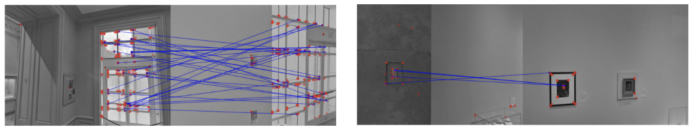

Fig. 7. Mismatch of door and corner

The possible reason for these situations is that SIFT relies too much on the gradient direction of the local area pixels in the main direction phase, which may make the main direction found inaccurate. The subsequent feature vector extraction and matching are heavily dependent on the main direction. It may cause the amplification error of the subsequent feature matching, and thus the matching is unsuccessful. Moreover, SIFT is an algorithm that only uses grayscale properties and ignores color information, which may also cause some deviations in image matching.

For the future improvement, some ideas that can be used to refine and optimize the model are:

- When matching the characteristics of the training image and the test image, not only the relationship between the test image and an image is considered, but other images that are similar to the test image content or consistent with the training image coordinates can be comprehensively considered to predict the coordinates.
- Get more accurate coordinates of the image by solving the essential matrix. We have tried this method before, and successfully calculated to decompose the essential matrix to get r and t, but there are still doubts about how to find the coordinates next, plus there is not enough time.
- Thinking about how to solve the problem that some walls cannot find matching images, a feasible method that comes to mind is to convert the picture into different gray values according to the color of the image, so as to distinguish the walls of different colors and make it easier to match.

## VII. Conclusion

Through experiments, it can be found that SIFT has a strong ability in image matching. In the case of a particularly large amount of data, FlannedMatch can be used to cooperate with SIFT to complete image similarity matching. Appropriate image filtering, such as Hash can help improve efficiency, and dHash is a good choice.

## REFERENCES


[1] H. Sarohi, "Image Retrieval using Perceptual Hashing", IOSR Journal of Computer Engineering, vol. 9, no. 1, pp. 38-40, 2013. Available: 10.9790/0661-0913840.

[2] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, vol. 60, no. 2, pp. 91-110, 2004. Available: 10.1023/b:visi.0000029664.99615.94.

[3] K. Neal, "Looks Like It", The Hacker Factor Blog, 2011. [Accessed 22 October 2021].

[4] A. Neelima and K. Singh, "Perceptual Hash Function based on Scale-Invariant Feature Transform and Singular Value Decomposition", The Computer Journal, vol. 59, no. 9, pp. 1275-1281, 2015. Available: 10.1093/comjnl/bxv079.

[5] S. Venkatraman, Arunaanand and Sharanya, "Real Time Mold Quality Inspection in Foundries using Image Processing Techniques," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-5, doi: 10.1109/ic-ETITE47903.2020.215.

[6] K. Sri, G. Manasa, G. Reddy, S. Bano and V. Trinadh, "Detecting Image Similarity Using SIFT", Expert Clouds and Applications, pp. 561-575, 2021. Available: 10.1007/978-981-16-2126-045 [Accessed 22 October 2021].