

國立臺灣大學電機資訊學院資訊網路與多媒體研究所

碩士論文

Graduate Institute of Networking and Multimedia

College of Electrical Engineering and Computer Science

National Taiwan University


Master Thesis

基於搜尋系統與快速空間關係驗證之

多物件辨識及定位

Multiple Object Localization by Search-based Object

Recognition and Fast Geometric Verification



林芳而

Fang-Erh Lin

指導教授：徐宏民教授

Advisor: Winston H. Hsu, Ph.D.

中華民國101年7月

July, 2012

誌謝

專題一年及研究所兩年的時間，不管是研究上、物質上、心靈上，要感謝非常多人給我的幫助與鼓勵。首先要感謝的是指導教授徐宏民老師。老師帶領我進入做研究的殿堂，教會我們做研究的方法與態度、鼓勵我們投paper參加conference、實驗結果因為程式有錯誤時安慰我們他也曾犯過一樣的錯。感謝kuonini學姊對我研究上的指導，總是在實驗室跟我們一起討論想法、找bug、提供各種程式上的協助，也帶領我們參加國際會議、協助修改paper及海報。感謝KT學姊總是給我們信心與鼓勵，不厭其煩的陪我們rehearsal、給予建議，是MiRA組的精神領袖！感謝安容、小光、玉米、佑家、小大頭一路在研究上的扶持：謝謝安容在合作project時認真嚴謹的態度讓我有所成長、謝謝小光總是熱心跟我一起想演算法及遇到的難題、謝謝玉米總是聽我抱怨及給我安慰鼓勵、謝謝佑家一起在server team扶持鼓勵面對緊張的研究進度、謝謝小大頭陪我聊天給我信心。感謝實驗室學長姊、學弟妹給予我研究上的幫助，很開心能夠認識大家，一起在歡樂的環境下做研究，我會很珍惜我們在MiRA的日子。感謝大學同學多年來的陪伴，雖然大家都不同實驗室，總還是能抽空一起聚餐分享研究生活的老梗、歡樂與淚水；謝謝呂敏總是特別撥冗幫我修改paper、謝謝丸偉當兵的時候不忘關心我的研究進度、謝謝小白摳拉繼續製造笑料、謝謝tony就坐在505讓我隨時能去串門聊天。另外要感謝親愛的黃彥傑一起分享我的喜怒哀樂，在我焦躁的時候承受我的壞心情、遇到難題的時候陪我一起想辦法解決、開心的時候一起慶祝、鼓勵我勇敢大步邁進。最後要感謝的是我重要的家人，體諒我因為忙碌很少返鄉、無條件支持我做想做的事讓我沒有後顧之憂。謝謝你們！

摘要

由於廣大的應用範圍及講求高效率的執行速度，近年來多物件的辨識與定位成爲一個重要的問題。早期的研究只有在小型的資料庫中搜尋，因此能在短時間內搜尋與定位；我們希望除了能夠在龐大的物件圖片資料庫中準確辨識目標圖片中的物件與標出位置，也能夠在短時間內完成動作。這篇論文中，我們提出兩種演算法Adaptive Window Search和Hierarchical Cluster Search，利用物件辨識系統對目標圖片進行多物件的搜尋與定位，也提出一個加速演算法FastGV以減短物件定位的時間。實驗結果顯示我們提出的演算法在多物件的辨識與定位有很高的準確率，同時有效縮短在大型物件資料庫中的搜尋與定位時間。

關鍵字：多物件辨識、多物件搜尋、物件定位、關係驗證



Abstract

Multiple object localization and recognition has been an important problem in recent years not only because of its difficulty to be time efficient but also due to many different schemes of widespread applications. In many previous works, only a limited amount of object models contribute to less computational time. However, they tend to not work efficiently together with large-scale database. In this paper, we propose two search algorithm and search-based object recognition system to recognize and localize multiple objects in an image with a large-scale database. Since we tackle this problem with the idea that users can get brief information of an item immediately after taking only a snapshot, a low response time is also taken into account. Therefore, we propose a new spatial verification algorithm to improve the speed of localizing objects. We implement the algorithms within a large-scale book recognition system and present experimental results that demonstrate the efficiency of our algorithms in terms of detection recall, precision, and speed compared to the baseline and efficient subwindow search (ESS) approaches.

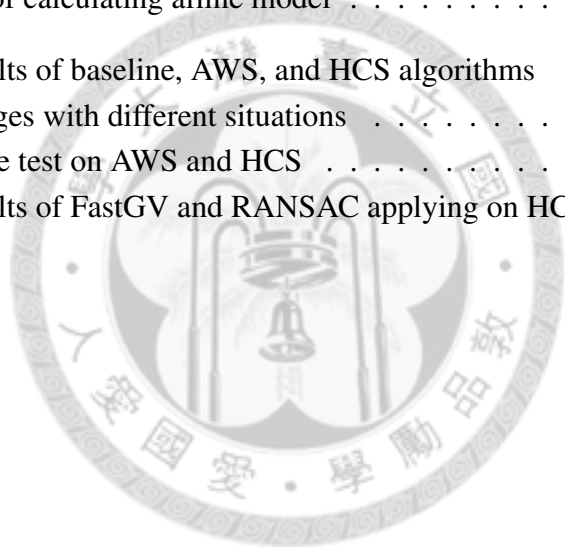
Keywords: Multiple object recognition, multiple object retrieval, object localization, geometric verification

Contents

誌謝	i
中文摘要	ii
英文摘要	iii
1 Introduction	1
2 Object Recognition System	5
2.1 Collecting Data	5
2.2 Recognition and Geometric Verification	5
3 Multiple Object Localization	7
3.1 Baseline Algorithm	7
3.2 Adaptive Window Search (AWS)	8
3.2.1 System Overview	8
3.2.2 Window Estimation	10
3.2.3 Grid-based Feature Density Matrix	11
3.3 Hierarchical Cluster Search (HCS)	11
3.3.1 Hierarchical Agglomerative Clustering	12
3.3.2 Top-Down Search	12
4 Fast Geometric Verification	15
4.1 Fast Geometric Re-Ranking for Image Retrieval	15
4.2 Fast Geometric Verification (FastGV)	15
4.2.1 Matching Feature Pairs	17
4.2.2 Geometric Similarity Scoring	17
4.2.3 Affine Model	18
5 Experiments and Discussions	19
5.1 Implementation Details	19
5.2 Dataset and Ground Truth	19
5.3 Baseline, AWS, and HCS	19
5.3.1 Recognition Results and Precision	21
5.3.2 Speed	23
5.4 RANSAC and Fast Geometric Verification	24
5.4.1 Recognition Results and Precision	26
5.4.2 Speed	26
5.5 Comparisons with Similar Works	26
6 Conclusions	28
References	29

圖目錄

1.1	Illustration of an application scenario	2
3.1	System diagram of adaptive window search	9
3.2	Flow chart of adaptive window search	10
3.3	Illustration of building hierarchical agglomerative cluster tree on query image	13
4.1	Illustration of location geometric similarity scoring in fast geometric re-ranking method	16
4.2	Illustration of location and orientation geometric similarity scoring in FastGV method	16
4.3	Illustration of calculating affine model	18
5.1	Sample results of baseline, AWS, and HCS algorithms	20
5.2	Sample images with different situations	22
5.3	Rotate image test on AWS and HCS	23
5.4	Sample results of FastGV and RANSAC applying on HCS algorithm . . .	25



表目錄

5.1	Evaluations of baseline, AWS, and HCS	20
5.2	Evaluations of RANSAC and FastGV	25



Chapter 1 Introduction

With the increasing popularity and more and more types of sensors being equipped on smart phones, the number of applications implemented on smart phones has been growing fast recently. Also, thanks to the prevalence of high-bandwidth mobile networks, it has been promising to acquire related information from a large-scale database by taking a snapshot using mobile phones and then sending the image to remote server for object recognition. As Figure 1.1 shows, consider a customer in a bookstore who wants to know comments and evaluations for each book on the bookshelf. Furthermore, with social networks such as Facebook and aNobii [1], the customer may even be able to know which friends have read those books before. Using the systems demonstrated in [2] [3], the customer takes a snapshot of one book each time with the camera-phone. The camera-phone sends the query image over the mobile network to a remote server and waits for related data of the book returned from the server after the item is recognized. However, to view the comments of all the books on the bookshelf, the customer has to take many snapshots and spend time on communication between the camera-phone and the server. Under this situation, proposing an algorithm to localize and recognize multiple objects efficiently is necessary.

In recent years, multiple object localization and recognition has been an important problem. To localize objects efficiently, a branch-and-bound approach named Efficient Subwindow Search (ESS) [4] was proposed to decrease the computational cost and outperform the original sliding window method. The branch-and-bound scheme repeatedly decreases the possible regions from the input image to bound the desired object using visual word histograms. Recently, Tom Yeh et al. proposed a data-dependent multi-class

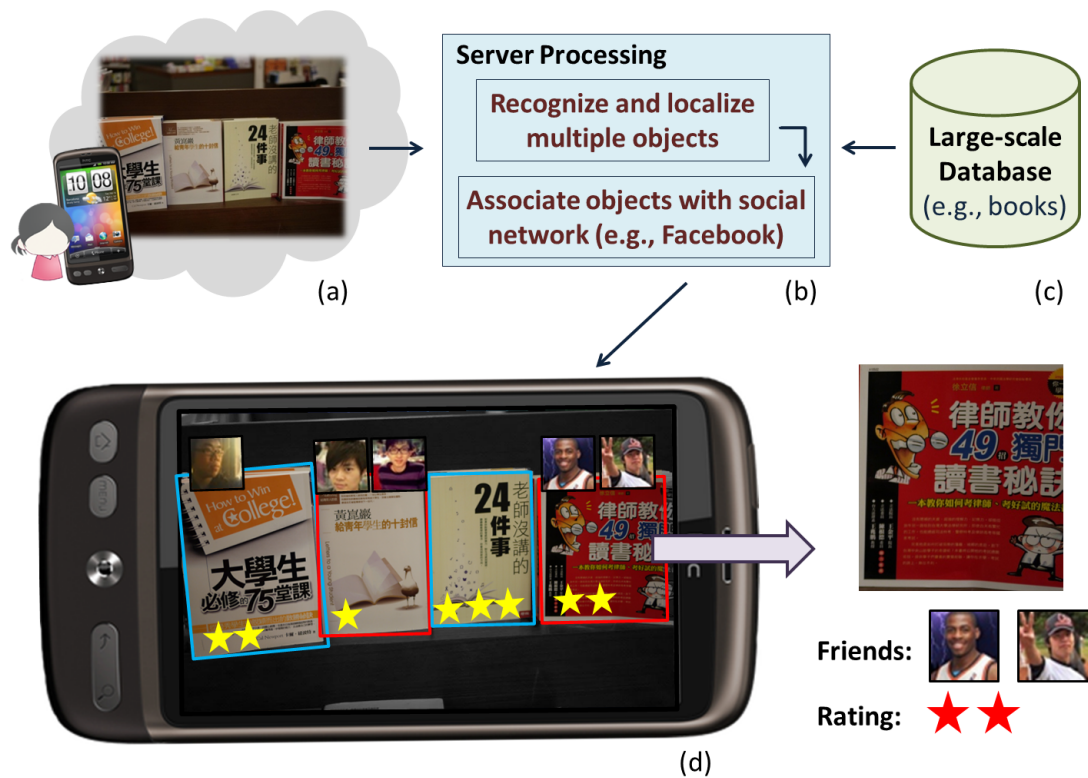


Figure 1.1: An application scenario of the multiple object localization and recognition problem. (a) Whenever users want to know the comments or other information of products in stores, they can take a photo by mobile phones and upload it to the remote server. (b) After recognizing and localizing multiple items, objects in the image can be detected and related information can be retrieved from social networks. (c) The recognition system is constructed by a large-scale database (e.g., million-scale object images). (d) Take books as example, users can see ratings of books and those friends who have read the books, by mobile phones.

branch-and-bound formalism that bounds the region of interest based on features' positions [5]. The authors in [6] proposed a Steiner tree model to solve multiple object detection and localization problem. This method selects windows most likely to contain the objects of interest by segmenting the entire image and post-processing such as merge and trim. By Steiner tree model, it selects regions and detects objects efficiently. These methods above all improve the speed to localize objects; however, for problems that contain larger-scale (i.e., millions of) object models in the database, such methods all fail to achieve real-time processing due to scalability issues.

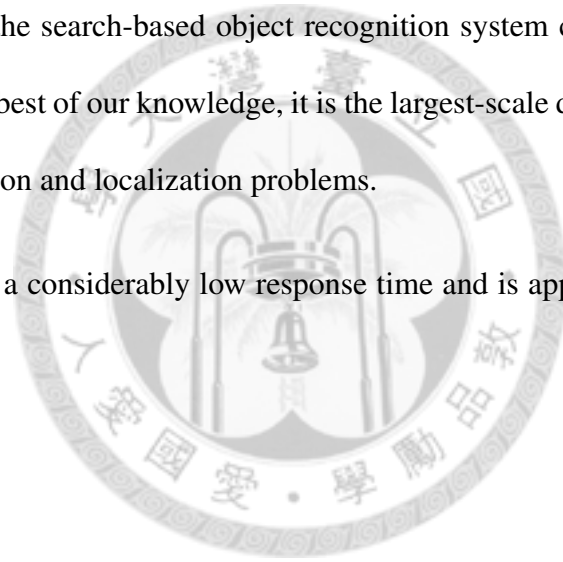
Geometric verification is an important issue in image retrieval problem. Usually we apply random sample consensus algorithm, also known as RANSAC [7], to verify the similarity of images. The idea of RANSAC is calculating the affine model from three randomly selected feature-matching pairs, and then counting the number of inlier agreements from other feature-matching pairs. By repeating this process many times, the affine model with most number of agreements is identified as the transformation model between the query image and the retrieved similar image. The number of agreements is also taken as a variable to check if the similar image has enough inlier matching features with the query image. Although the computation time of RANSAC is short enough to be applied in real-time system, it still takes some time to guess the appropriate affine model. Sam S. Tsai [8] proposed three geometric scoring methods improving the speed to find geometric consistency and re-rank the similar image list. However, they did not combine the three scores to re-rank the list but only compared the results of them.

Through these previous works [5] [6], it is obvious that multiple object recognition and localization problem is important but not addressed properly. Meanwhile, improv-

ing the geometric verification step can effectively shorten the time of object localization. Therefore, we design an algorithm to efficiently localize multiple objects with a large number of object models and recognize objects by a search-based recognition system.

To summarize, the main contributions of this work are:

- Proposing two methods to solve the multiple object localization problem by a search-based object recognition system.
- Proposing an algorithm to speed up the object localization step.
- A database of the search-based object recognition system constructed of 251,000 images. To the best of our knowledge, it is the largest-scale database in the multiple object recognition and localization problems.
- A scheme with a considerably low response time and is applicable in mobile augmented reality.



Chapter 2 Object Recognition System

2.1 Collecting Data

In the multiple object localization problem, we start from multiple book localization due to the application scheme and the easier data collection. In order to collect a sufficient amount of book images, data from a large book selling website [9] is crawled. The meta-data of each book (title, ISBN, price and comments) is fetched as well. There are no websites collecting book spines. Therefore, book images in our database only consist of book covers.

2.2 Recognition and Geometric Verification

After the image data is collected, a bag-of-feature-points is extracted from each book image. We have tried different local image detectors such as Harris-affine [10], Hessian-affine [11] and Maximally Stable Extremal Region [12], and different descriptors such as SIFT [13] and SURF [14] for better recognition accuracy. Finally we adopted Hessian-affine detector and SIFT descriptor. For fast searching through a large-scale database, each book image is quantized to a set of visual words through a vocabulary tree. Because of the superiority in efficiency of inverted indexing, we retrieve books with similar visual word histograms efficiently.

Because different books usually have dissimilar covers and users only care about the images exactly shown in the query image, in the book recognition process we only focus on N candidate images (e.g., the five best-matching images) and perform a geometric

consistency check to find the correct matching book. We propose a spatial verification algorithm to find spatial consistency between the features of the query image and candidate images. The candidate image with the largest and above-a-threshold number of matching inliers is chosen as the correct matching book, and will be returned to the multiple object recognition process. Moreover, an affine model from the correct matching book in the database to the query image is generated by matching inliers. It enables us to bound the region of the book in query image.



Chapter 3 Multiple Object Localization

There are some common characteristics of book display on bookshelves in bookstores. Books in the same bookshelf are often put densely together. However, due to the complexity and variety of covers, the approach to segment covers of different books does not work as in [15]. It is hard to find main edges between books by edge detection and get the dominant direction using Hough transform. Therefore, we design a context-aware adaptive window search method, which proposes regions by already matched books in a row-based manner (Section 3.2.2), and a grid-based feature density matrix for supporting information (Section 3.2.3). Due to the fact that objects are not always placed orderly, we propose another algorithm based on clustering features into several groups to do the recognition and localization processes. Details are described in Section 3.3.

3.1 Baseline Algorithm

To the best of our knowledge, there has been no similar work that uses search-based recognition system in large-scale database to detect multiple objects. Therefore, we propose a simple algorithm as the baseline method. The recognition result of the input image may be influenced by noisy visual words such as features of bookshelves. However, it is still possible to get book images which show up in the input image if more similar images are selected and checked by geometric verification. First we send a request to the recognition server that recognizes the whole input image, and perform a spatial verification for the top N (e.g., 10) ranked similar images. After selecting the best-matching image, we remove the features in the matched region from original input image and send a new request. By

repeating these steps, features sent to the recognition server become less and the recognition of other books will not be easily affected by visual words of already-recognized books, which pose as noise for recognizing remaining books. Therefore, we can retrieve all the matching books after repeating requests many times.

3.2 Adaptive Window Search (AWS)

3.2.1 System Overview

Figure 3.1 provides an overview of AWS system, which is mainly divided into two parts, the adaptive window search process and the object recognition process. For the object recognition part, we have downloaded a large amount of object images from a product selling website (Section 2.1) and constructed an online query system (Section 2.2). In order to locate the informative regions in the query image, the grid-based feature density matrix is calculated and provided to find the candidate regions which may contain objects. First we query the input image by the recognition system, and get the initial region using geometric verification to check feature matching consistency. The adaptive window algorithm proposes a region of interest relying on information provided by the initial region or the feature density matrix. Then, images similar to the proposed region will be retrieved and checked for spatial consistency to find matched objects. By repeating these steps, objects in the input image will be localized and recognized gradually.

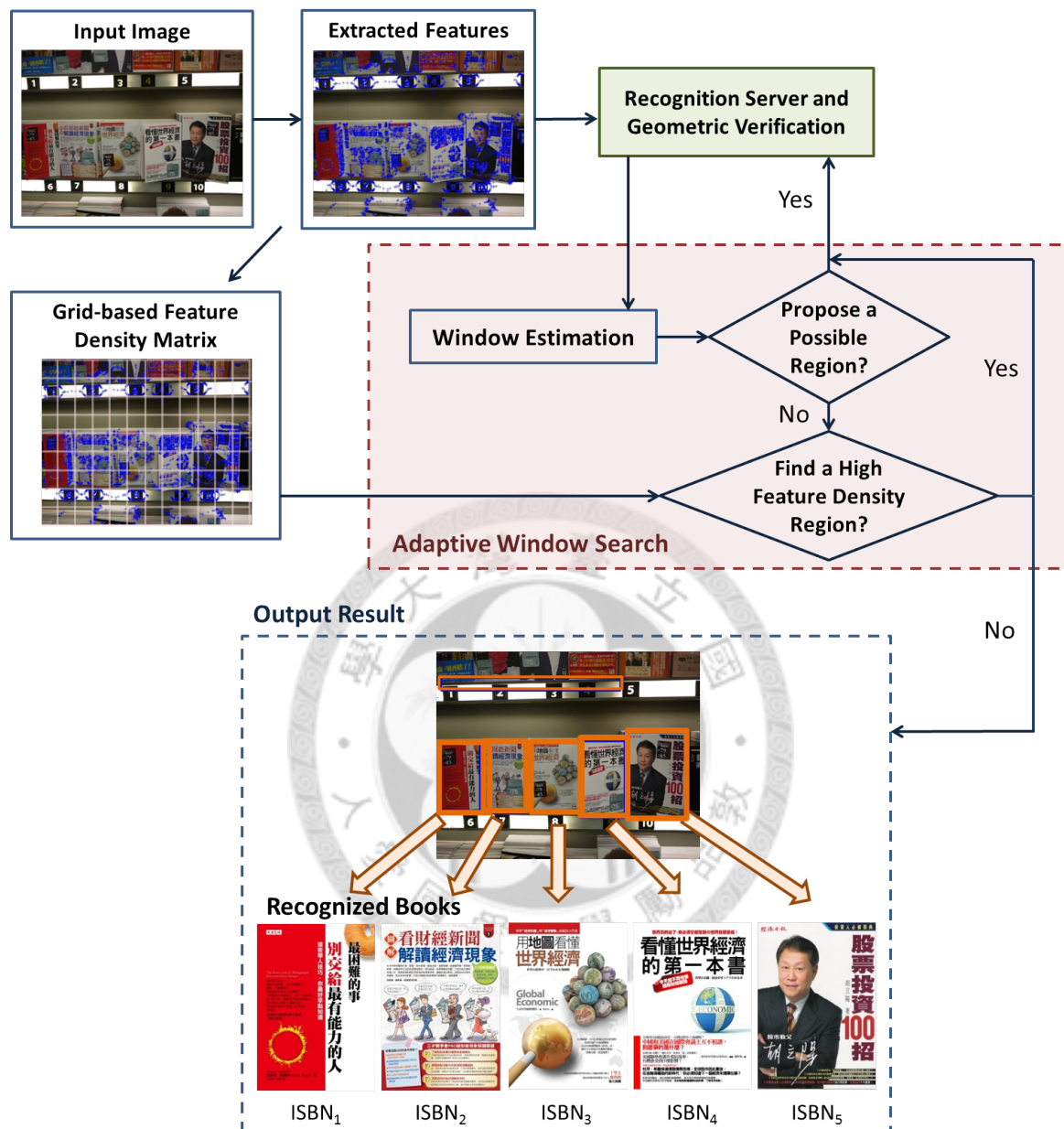


Figure 3.1: System diagram. Once the image is sent to the server, its features will be extracted and used to calculate a grid-based feature density matrix. At first we retrieve similar images for the input image and verify the spatial consistency between the features of input image and images in database, in order to find an initial region containing the book. Then we find other regions of interest by adaptive window search, integrating the information of the initial region and the feature density matrix. The result is output if there is no other high feature density region to retrieve. ISBNs for each recognized book enable us to retrieve other information (e.g., brief introduction, price, rating, comments) through social networks (e.g., Facebook, aNobii).

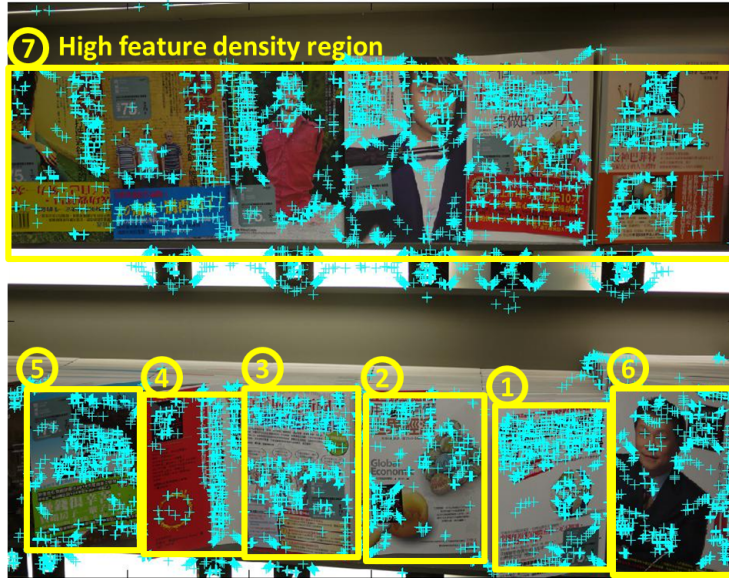


Figure 3.2: Flow chart of adaptive window search. The number on each mask denotes the sequence number. No. 1 is the initial region after querying the input image and that spatial consistency is verified. Regions No. 2 ~ 6 are proposed by window estimation and verified through the recognition system. After scanning all the adjacent areas, No. 7 is recommended via feature density matrix, and then previous steps are repeated. Cyan marks are extracted features of the image.

3.2.2 Window Estimation

Empirically, although visual features of all the other books in the input image represent background clutter when we focus on a specific book, it is still possible to get the best-matching image, which is one of the books in the input image, returned from the recognition system, if the query image is the whole input image. Therefore, we assume that the best-matching image after geometric verification, as returned from the recognition system, is one of the books in the input image (e.g., No. 1 in Figure 3.2). We retrieve not only the affine model of the best-matching book but also the approximate size of the book. Due to the fact that all books are about the same size and that books near each other are always packed together, we can estimate the size and position of the books near the currently recognized one (e.g., No. 2 or 6 in Figure 3.2). The estimated region is sent to the recognition system for retrieving the answer that whether there is any book in this

region. Only at most five best-matching books will be checked by spatial consistency to find the best-matching result. The affine model will also be adjusted to the correct region that contains books, and then the revised region and size of the newly recognized book are used to propose the next assumption. The process stops when the remaining region is too small to recognize a book.

3.2.3 Grid-based Feature Density Matrix

The method described in Section 3.2.2 can only localize objects in the same row if one of those books is recognized first as shown in Figure 3.2 (No.2 ~ 6). For other books not recognized and localized yet, we adopt another observed characteristic in bookstores to decrease the time spent in selecting query regions and reduce the amount of features to process. The vacancy between different rows of bookshelves usually contains few features while the covers of books can have a large amount of features extracted, as shown in Figure 3.2. A grid-based feature density matrix can be computed to measure where the vacant space may be and where books show up. If the window search stops because of getting close to a border, we check the density matrix to find another high density region to start up the sliding window process again until the density of the remaining region is smaller than a threshold.

3.3 Hierarchical Cluster Search (HCS)

We observed that all books are placed orderly and densely in the bookstore; therefore, Adaptive Window Search works under this situation. In the condition that objects are not always in the same size and are not placed orderly, we development an algorithm, called

Hierarchical Cluster Search, without any assumption.

The intuition to solve the multiple object localization problem is segmenting the objects in the query image into different sub-images. However, the number of objects in the query image is unknown. It is hard to decide the number of sub-images being clustered. Here we adopted Hierarchical Agglomerative Clustering method, which will be illustrated in Section 3.3.1, to build a hierarchical cluster tree with different cluster number in each level. Therefore, the clusters in each level are treated as an object and can be sent to the object recognition system to localize the exact position. The details will be described in Section 3.3.2.

3.3.1 Hierarchical Agglomerative Clustering

In hierarchical agglomerative clustering, every feature point is treated as a cluster at first. The second step is selecting two clusters which have the smallest Euclidean distances, forming a new cluster, and calculating its new center by average Euclidean distance. Then, the second step is repeated until all the clusters are grouped into the same cluster. Then we can choose several thresholds cutting the tree to get different cluster numbers of each level. Thus we get different segmented results even the number of objects in the query image is unknown. As shown in Figure 3.3, we build a hierarchical tree by applying hierarchical agglomerative clustering method on the query image.

3.3.2 Top-Down Search

After getting a cluster tree, we send clusters of each level to the object recognition system for retrieval and geometric verification through top to down approach. If the object is



Figure 3.3: An example of applying hierarchical agglomerative clustering on the query image. There are four levels within different number of clusters. In each level, feature points with the same color represent they are in the same cluster.

retrieved and localized, it will be removed in the cluster in the lower level. Thus it will not be recognized again in the following cluster recognition process. By segmenting the query image into clusters, we effectively decrease the noise which will affect the query result in object recognition system. Therefore, it is much possible to get the correct object showing up in the top N (e.g., 5) query results.

The reason we adopted top-down search instead of bottom-up search is that we have less clusters we need to search with the former approach. If we start searching from the bottom level, the number of clusters we need to scan to detect the same number of objects is probably more than from the top level. Therefore, adopting top-down search is effective than bottom-up search.



Chapter 4 Fast Geometric Verification

4.1 Fast Geometric Re-Ranking for Image Retrieval

Sam S. Tsai [8] proposed a method improving the speed to find the geometric consistency between query image and retrieval results, and to re-rank the result list by three geometric similarity scoring methods: location, orientation, and scale [8]. As shown in Figure 4.1 [8], the geometric consistency can be found through calculating three scores from a matching feature pair list M . S_{LDR} is the location score, S_{OD} is the orientation score, while S_{LSR} represents score of scale. $dist(.,.)$ describes the Euclidean distance of two points.

$$S_{LDR} = \left\{ \log\left(\frac{dist(l_{q,i}, l_{q,m})}{dist(l_{d,j}, l_{d,n})}\right) \mid (i, j), (m, n) \in M \right\} \quad (4.1)$$

$$S_{OD} = \left\{ (o_{q,i} - o_{d,j}) \mid (i, j) \in M \right\} \quad (4.2)$$

$$S_{LSR} = \left\{ \log\left(\frac{s_{q,i}}{s_{d,j}}\right) \mid (i, j) \in M \right\} \quad (4.3)$$

4.2 Fast Geometric Verification (FastGV)

In the previous method, they compared the re-ranked result lists applied by three scoring methods but did not combine them to get a new re-ranked list. Therefore, we propose a fast geometric verification method to modify the scoring method and calculate the trans-

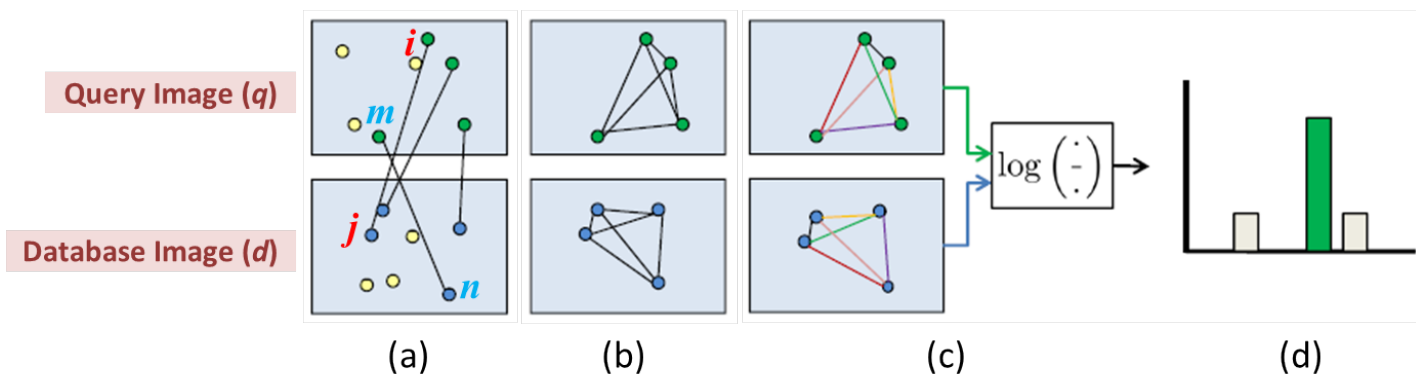


Figure 4.1: The steps of generating location geometric similarity score: (a) features in the query image and database image are matched according to the visual word ID, (b) pairwise feature distances are calculated, (c) log distance ratios of the corresponding feature pairs are calculated, and (d) log ratio histogram. The maximum value of histogram is the location geometric similarity score.

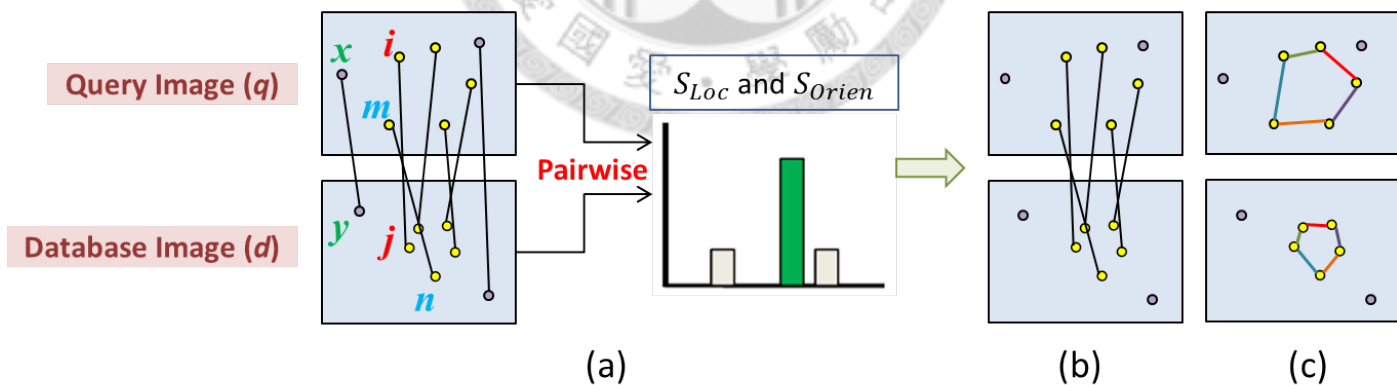


Figure 4.2: The steps of preserving inlier feature matching pairs: (a) calculate the location and orientation scores for each feature matching pair and generate histograms correspondingly, (b) preserve the feature matching pairs which are simultaneously agreed with the largest number of bins in location and orientation scoring histogram, (c) the affine model can be calculated after removing outliers (incorrect matching pairs).

formation model. As shown in Figure 4.2, the main idea of fast geometric verification is finding the consistency of rotation angle and scaling ratio for the matching feature pairs.

4.2.1 Matching Feature Pairs

In the object retrieval step, the similarity is calculated by the number of visual word matching pairs. Therefore, there are several feature matching pairs with the same visual word ID in each query image and retrieved image pairs. We calculate the location and orientation similarity scores through these feature matching pairs.

4.2.2 Geometric Similarity Scoring

The location geometric similarity scoring describes the scale ratio between the query image and the result image in database. q_i, q_m are the features in the query image; d_j, d_n are the corresponding features in database image, which is retrieved by recognition server. Therefore, we propose to calculate the scale ration as

$$S_{Loc} = \left\{ \log \left(\frac{|\vec{q}_i \vec{q}_m|}{|\vec{d}_j \vec{d}_n|} \right) \mid (i, j), (m, n) \in M \right\} \quad (4.4)$$

Different from [8], we propose to calculate the angle between two pairs. The orientation geometric similarity scoring calculates the rotation angle between $q_i q_m$ and $d_j d_n$.

$$S_{Orien} = \left\{ \frac{\vec{q}_i \vec{q}_m \cdot \vec{d}_j \vec{d}_n}{|\vec{q}_i \vec{q}_m| |\vec{d}_j \vec{d}_n|} \mid (i, j), (m, n) \in M \right\} \quad (4.5)$$

We then estimate the number of features that have similar scaling ratio and similar rotation angle by generating two histograms correspondingly. To remove outliers, we



Figure 4.3: The steps of getting affine model: (a) original feature matching pairs after object recognition system, (b) counting a grid-based feature density matrix after removing incorrect feature matching pairs, (c) calculate affine model by three random features which are from three different high density grids.

keep the matching features which are simultaneously agreed with the largest number of bins in location and orientation scoring histograms. In this way, we have considered the location and orientation geometric similar scoring at the same time.

4.2.3 Affine Model

To efficiently obtain the transformation model after removing outlier matching features, we generate a grid-based feature density matrix and pick three random features from three different grids with high feature density to calculate affine transform as shown in Figure 4.3. The reason to pick features from different high density grid is to make sure the feature points we choose will not be too close to each other and result in a bias of the affine model.

Chapter 5 Experiments and Discussions

5.1 Implementation Details

The recognition system and feature extraction for input images are implemented in C code in order to fulfill real-time retrieval. We chose MATLAB to develop geometric verification process, adaptive window search, and hierarchical cluster search algorithm for easier matrix operation and convenient image processing.

5.2 Dataset and Ground Truth

To construct a large-scale database, we use a set of over 251,000 book images crawled from [9]. The average dimension for each image is 200 x 280 pixels. Hessian-affine and SIFT features are extracted from each book image. A set of features of randomly selected 20 thousand images is used to train a hierarchical vocabulary tree with 6 levels and 10^6 leaf nodes. Note that we adopt this approach on Oxford buildings dataset and find it is comparable with the baseline reported in [16]. For the testing dataset, we take 27 photos containing 155 books in the bookstore. The localization is considered positive if its intersection with a ground truth bounding box divided by their union is greater than 80% and the recognition result is correct.

5.3 Baseline, AWS, and HCS

The results of baseline, AWS, and HCS are shown in 5.1. As shown in Table 5.1, only 47.1% books are detected by the baseline (RANSAC) algorithm but 90.12% books are

Table 5.1: Results for recall, precision and speed by applying baseline, AWS, and HCS algorithms. Oracle result is performed by recognizing ground truth books.

Algorithm	Oracle	Baseline (ESS)	Baseline (RANSAC)	AWS (RANSAC)	HCS (RANSAC)
Testing Image	27				
Total Books	155				
Detected Books	X	43	81	111	96
Recognized Books	121	34	73	101	84
Recall	78.06%	21.94%	47.10%	65.16%	54.19%
Precision	X	79.07%	90.12%	90.99%	87.5%
Processing Time per Book	X	90.94s	19.95s	4.42s	14.58s

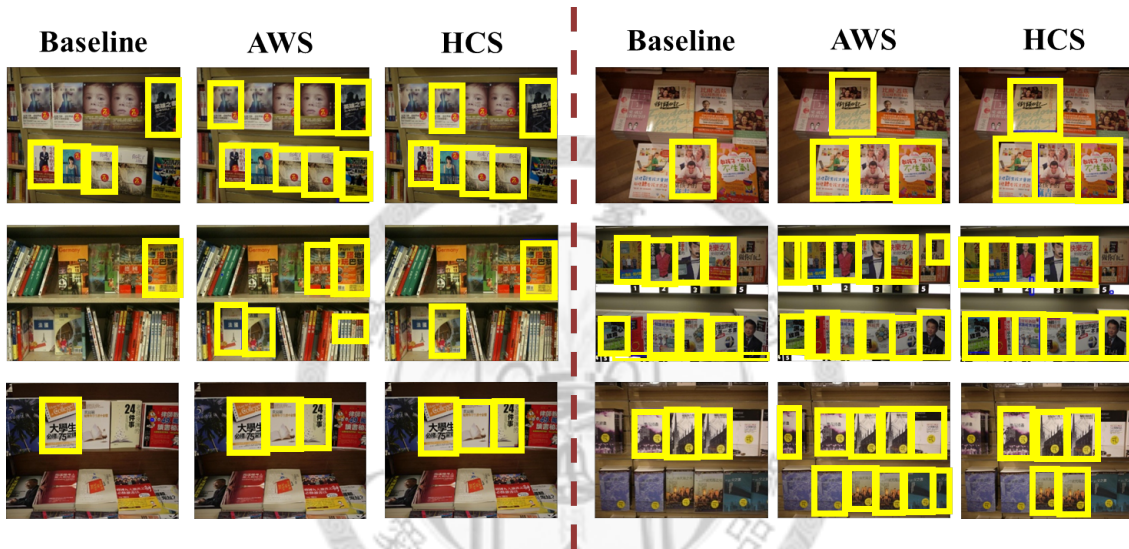


Figure 5.1: Sample results of baseline, AWS, and HCS algorithms. Yellow bounding boxes represents the books recognized and localized by the system.

recognized correctly, while 65.16% books are detected and 90.99% recognized correctly by adaptive window search (AWS), and 54.19% books are detected and 87.5% recognized correctly by hierarchical cluster search (HCS). The localization results are shown in Figure 5.1. The processing time shown in last row of Table 5.1 is the average of overall time spent on recognition and geometric verification for all testing images divided by the sum of recognized books. Average processing time for each book is 19.95 seconds by baseline, 4.42 seconds by AWS, and 14.58 seconds by HCS.

5.3.1 Recognition Results and Precision

As shown in Table 5.1, adaptive window search and hierarchical cluster search perform better than the baseline method by ignoring the cluttered background features and focusing on the feature matching in the proposed region or cluster. However, the book images we took in the bookstore are affected by lighting; also, sometimes the books are wrapped in an extra cover, which makes them look differently from images in the database, and may cause true negatives. There is another situation that the cover of a book is mild with few features, resulting in that matching inliers will be too few to pass the threshold. Moreover, the layout of books on bookshelves in the bookstore may not be orderly; for example, one of the books is placed vertically while the book next to it is laid down, or books with covers revealed are put together with others with spines laid outside. For above situations (as shown in Figure 5.2), achieving high recall in book localization is very challenging. Oracle testing results shown in Table 5.1 prove the difficulty recognizing books in bookstore with situations mentioned above. Only 78.06% of ground truth books can be recognized by the recognition system. Compared to oracle testing results, AWS achieves 83.47% recognition recall and HCS achieves 69.42% recognition recall. With the help of geometric verification, the books are almost recognized correctly in the detected region with a high precision rate.

Comparing AWS with HCS, we find that AWS performs better than HCS in both recall and precision because the books in testing data we collected are all placed orderly and vertically. In this way, AWS can recognize and localize books correctly and efficiently. However, the hierarchical agglomerative clustering process in HCS cannot separate books in each cluster properly. It sometimes separates a book into different clusters. Therefore,



Figure 5.2: Sample images with different situations. (a) The same book with slight differences as in the bookstore and in the database. The left ones are taken in the bookstore with wrappers on the bottom of the cover. The right ones are images in database. (b) The sample book with a clear cover having only a small number of features. (c) The book image in the database with a poor resolution. (d) Images taken in the book-store with an irregular layout. The upper image shows that some books are laid down while others stand up. The books in the bottom image show the fact that they may have different displaying orientations even when on the same bookshelf.

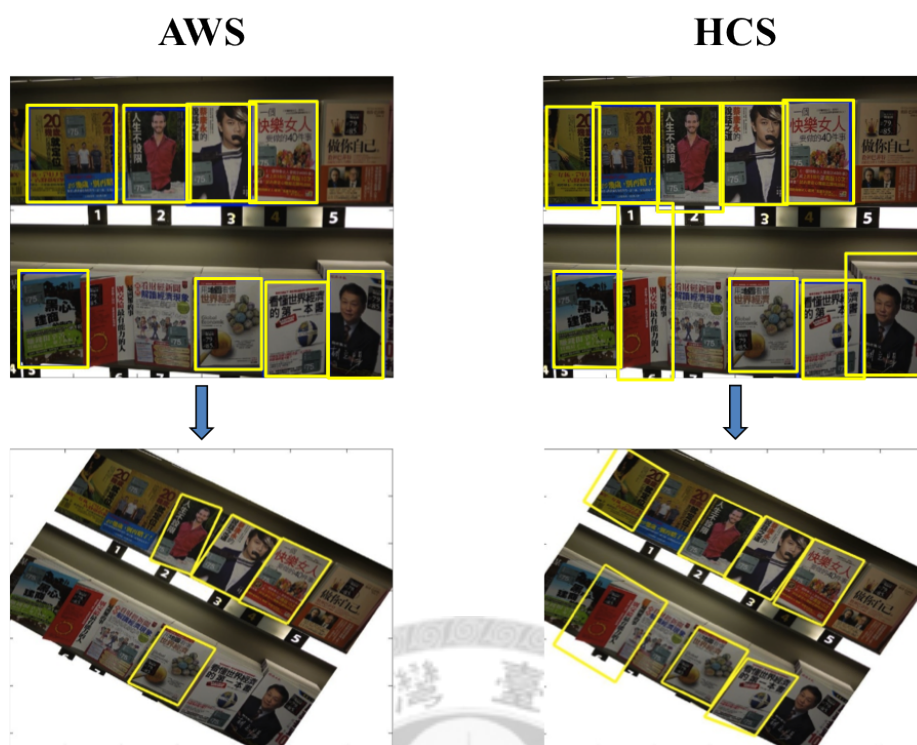


Figure 5.3: AWS and HCS are applied on an original image and a rotated image. There are 8 books recognized and localized correctly by AWS originally but only 4 books retrieved after we rotate the image. There are 10 books obtained by HCS and 7 books are retrieved afterwards.

AWS get better recall and precision than HCS.

To show the superiority of HCS, we rotate a testing image and applied on AWS and HCS as shown in Figure 5.3. There are 8 books recognized and localized correctly by AWS originally but only 4 books retrieved after we rotate the image. On the other hand, there are 10 books recognized and localized by HCS originally and 7 books are retrieved afterwards. It proves that HCS can deal with the images with unorganized objects.

5.3.2 Speed

Localization and recognition in adaptive window search algorithm is four times faster than the baseline (RANSAC). The main reason why the baseline method takes such a long time

to localize and recognize each is that there are more book images that have to be checked by geometric verification for each round than AWS. Besides, there are also more features in each recognition step because all the features in input image have to be seen except for those in matched regions. Nevertheless, with AWS, the number of features sent to the recognition server is smaller than baseline because only those in the proposed region are sent. The proposed region contains fewer noisy features, and the feature-matching step has less chance to be affected by the cluttered background. Therefore, we only have to care about the first image matching result, on whether the number of its inlier features is large enough. Also, empirically, the number of images that need to be checked by spatial verification can also be reduced to at most five. For these reasons, AWS does decrease the time to recognize and localize objects.

HCS does not improve a lot compared with the baseline algorithm in the speed part. HCS needs to spend time building the hierarchical clustering tree, and send every cluster in each level to the recognition server. The number of features in each cluster is fewer than the whole image sent to server in the baseline but larger than the probable region in AWS, which is the reason why HCS is faster than baseline but slower than AWS. Also, AWS stops searching possible region effectively; however, HCS checks every cluster in every level until the number of inliers in a level is smaller than the threshold. It causes HCS to spend more time than AWS.

5.4 RANSAC and Fast Geometric Verification

To evaluate the effectiveness of fast geometric verification (FastGV), we conduct the experiments, and the experimental results by applying FastGV and RANSAC on the HCS

Table 5.2: Results for recall, precision and speed by applying RANSAC and FastGV on AWS and HCS algorithms. Oracle result is performed by recognizing ground truth books.

Algorithm	Oracle	AWS (RANSAC)	AWS (FastGV)	HCS (RANSAC)	HCS (FastGV)
Testing Image		27			
Total Books		155			
Detected Books	X	111	85	96	88
Recognized Books	121	101	84	84	81
Recall	78.06%	65.16%	54.19%	54.19%	52.26%
Precision	X	90.99%	98.82%	87.5%	92.05%
Processing Time per Book	X	4.42s	4.22s	14.58s	10.73s



Figure 5.4: Sample results of FastGV and RANSAC applying on the HCS algorithm. Yellow bounding boxes represent the books recognized and localized by the system.

algorithm are shown in Figure 5.4. As shown in Table 5.2, RANSAC has better recall than FastGV in both AWS and HCS algorithms, but has lower precision and lower speed. In AWS algorithm, RANSAC gets 65.16% recall and 90.99% precision while FastGV gets 54.19% recall and 98.82% precision. RANSAC spends 4.42 seconds to recognize and localize a book while FastGV spends 4.22 seconds. In HCS, RANSAC gets 54.19% recall and 87.5% precision while FastGV gets 52.26% recall and 92.05% precision. RANSAC spends 14.58 seconds to recognize and localize a book while FastGV spends only 10.73 seconds. It shows that the proposed FastGV algorithm can achieve similar performance but reduce the processing time.

5.4.1 Recognition Results and Precision

All feature matching pairs will be considered to estimate the affine model by RANSAC algorithm, while FastGV algorithm checks the possible feature matching pairs first for geometric consistency and then only the matching pairs in the high inlier density region will be taken into account to calculate the affine model. Under this situation, some possible inliers (correct matching feature pairs) will be filtered and some objects cannot be localized by FastGV. RANSAC achieves higher recall than FastGV. On the other hand, RANSAC is much likely to get false positive, taking an outlier feature matching pairs as an inlier. Therefore, FastGV performs better than RANSAC in precision.

5.4.2 Speed

RANSAC algorithm proposes a possible affine model in each round, and all the feature-matching pairs need to vote for each round. In this way, FastGV is faster than RANSAC because feature-matching pairs are checked only once to calculate the three geometric similarity scores. The geometric verification step in AWS spends 0.4s by RANSAC and 0.33s by FastGV. It proves that FastGV saves around 17.5% time than RANSAC.

5.5 Comparisons with Similar Works

We adopt ESS [4] as a geometric verification method in baseline algorithm. The recall rate is 21.94% and the precision is 79.07%, and it takes 90.04 seconds for each book (as shown in Table 5.1). The localization results by ESS are worse than by RANSAC. Therefore, it is difficult to propose a probable region that contains book and it takes longer time to

recognize and localize books. Considering the worst case that there are no books in the input image, branch-and-bound method in [4] [5] may take a large amount of iterations to converge to the optimal solution. By AWS, if the best-matching image retrieved from the recognition server for the input image does not pass the threshold after geometric verification, the process automatically stops proposing any region of interest. Therefore, the way of AWS handling the worst case is more efficient than previous works.



Chapter 6 Conclusions

In this work, we have proposed two algorithms, adaptive window search and hierarchical cluster search, to select possible regions that contain books and send them to the recognition server for image matching and geometric verification. We have also proposed a fast geometric verification algorithm to improve the speed of the verification step. The results show that we did not get significant improvement in HCS due to the testing dataset, but with AWS, we obtain an improvement around 4 times in speed and 20% in recall. The fast geometric verification algorithm gets a speed improvement in both AWS and HCS. In AWS, the geometric verification time is shortened in 17.5% time. In future work, applications can be designed to use the information of books such as ISBN as the bridge to retrieve more related data through social networks after recognizing and localizing multiple books. AWS and HCS can also be applied to other recognition system with large-scale database (e.g., logos of daily objects) to develop other applications.

References

- [1] aNobii, “anobii,” <http://www.anobii.com/>.
- [2] Sam S. Tsai, David Chen, Vijay Chandrasekhar, Gabriel Takacs, Ngai-Man Cheung, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod, “Mobile product recognition,” in *Proceedings of the international conference on Multimedia*, New York, NY, USA, 2010, pp. 1587–1590, ACM.
- [3] Stephan Gammeter, Alexander Gassmann, Lukas Bossard, Till Quack, and Luc Van Gool, “Server-side object recognition and client-side object tracking for mobile augmented reality,” in *Proceedings of IEEE International Workshop on Mobile Vision (CVPR 2010)*, 2010.
- [4] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann, “Beyond sliding window: object localization by efficient subwindow search,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [5] Tom Yeh, John J. Lee, and Trevor Darrell, “Fast concurrent object localization and recognition,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 280–287.
- [6] Olga Russakovsky and Andrew Y. Ng, “A steiner tree approach to efficient object detection,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 1070–1077.

- [7] Martin A. Fischler and Robert C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, 1981.
- [8] Sam S. Tsai, David Chen, Gabriel Takacs, Vijay Chandrasekhar, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod, “Fast geometric re-ranking for image-based retrieval,” in *Proceedings of IEEE International Conference on Image Processing, 2010*, September 2010, pp. 1029–1032.
- [9] 博客來, “博客來,” <http://www.books.com.tw/>.
- [10] Krystian Mikolajczyk and Cordelia Schmid, “Scale and affine invariant interest point detectors,” *International Journal of Computer Vision*, vol. 60, pp. 63–86, 2004.
- [11] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, “A comparison of affine region detectors,” *International Journal of Computer Vision*, vol. 65, pp. 43–72, 2005.
- [12] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.
- [13] David G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, November 2004.
- [14] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, “Surf: Speeded up robust features,” in *Computer Vision – ECCV 2006*, Aleš Leonardis, Horst Bischof, and Axel

Pinz, Eds., vol. 3951 of *Lecture Notes in Computer Science*, pp. 404–417. Springer Berlin / Heidelberg, 2006.

[15] David Chen, Sam S. Tsai, Bernd Girod, Cheng-Hsin Hsu, Kyu-Han Kim, and Jatinder Pal Singh, “Building book inventories using smartphones,” in *Proceedings of the international conference on Multimedia*, New York, NY, USA, 2010, pp. 651–654, ACM.

[16] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman, “Object retrieval with large vocabularies and fast spatial matching,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

