POSITIONING-BASED IMAGE RETRIEVAL APPLICATION

[1] *Qiao Liang* (梁橋), [2] *Xiaobei Qian* (錢曉蓓), [1] *Chiou-Shann Fuh* (傅楸善)

[1] Graduate Institute of Networking and Multimedia,
National Taiwan University, Taiwan
[2] Graduate Institute of Electrical Engineering,
National Taiwan University, Taiwan
Email: clarenceliang@foxmail.com, joey.green9337@gmail.com, fuh@csie.ntu.edu.tw

ABSTRACT

In this paper, we present an image retrieval application for using a single query image to retrieve images taken at the same position and to know where it is taken. This application applies to the situations that one wants to see more pictures about a scene from a specific position with only one image at that position. Having pre-built the 3D model of the scene by using *structure from motion*, we match the query image with the 3D model to know where the image is taken. Then the positioning result will be used to find more images taken there efficiently by a proposed backward search upon the existing image collections. We have also designed and implemented a graphic user interface for this application, and tested it within the campus for real experiments.

Keywords: *Vision-based Positioning; Image Retrieval; Structure from Motion*.

1. INTRODUCTION

With the development of technology, there has been a rapid increase in the size of digital image collections on the Internet. However, it is hard to make use of the information unless we retrieve the images correctly and efficiently. Image retrieval has been a very active research area in the past decades. Many image retrieval systems have been developed. There are two main methods: text-based and content-based. The text-based approach can be tracked back to late 1970s. In this method, the images are manually annotated by text, which are then used by a database management system (DBMS) to perform image retrieval. Representatives of this approach are [3, 4]. Content-based image retrieval (CBIR) was introduced in the early 1980s. In CBIR, images are indexed by their visual content, such as color, texture, and shapes. Early representative works of this approach are QBIC [5], Photobook [6], VisualSEEK [7]. Moreover, there have been many great researches on this topic in recent years. Most traditional content-based image retrieval methods use the basic information of the images, such as color histograms, shapes, or different kinds of features. All the information helps us find the images that are most similar to the query image. However, there are still some situations, where these kinds of retrieval methods are not suitable for us to find the results we want. Consider one case, where we want to know where a photograph is taken, and retrieve more images taken in the same position. In such cases, we need to find out the implicit position information in the images, instead of some basic information. The differences between the results of these two kinds of image retrieval methods are shown in Fig. 1. The part (a) of the figure shows the retrieval results from traditional CBIR systems. We use one image taken in front of the NTU Main Library to retrieve other images that show the front side of the library. In this case, the kind of feature used for retrieval is the image content and the distance between two images is defined as the similarity of image contents. This kind of image retrieval helps find similar images. The part (b) of Fig. 1 shows the retrieval results from positioning-based image retrieval systems. In this case, we not only use the image content for retrieval, but also use it to figure out the relationship between the images in 3D space. The positions of two images are the used to compute the distance and to determine the similarity between these two images. As a result, we can find those images with closer position with the query image in 3D space, even though they may have little similarity in the direct image content.

(a) (b)

Fig. 1: (a) Traditional image retrieval. (b) Positioning-based image retrieval.

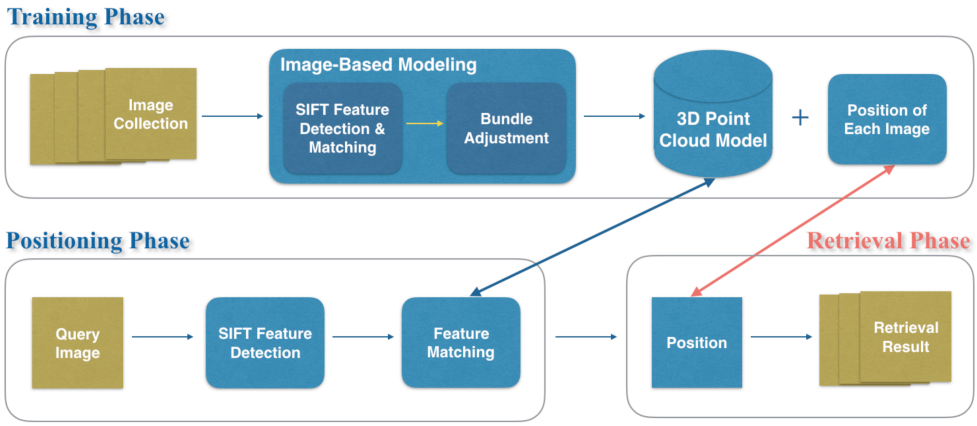

Fig. 2: Overview of the proposed positioning-based image retrieval system. The three main phases are shown in the figure. The yellow blocks denote the input and output in this system.

To achieve this purpose, there are three main phases: training phase, positioning phase, and retrieval phase. In the training phase, if we have an image collection of a specific scene or area (e.g., Internet searching results of the Taipei 101), we use the image collection to reconstruct the 3D point cloud of this scene. In the positioning phase, we match the feature points of the query image with the 3D point cloud to compute the position of the query image. In the retrieval phase, we use the positioning result to perform a backward searching and find the images closest to the positioning result from the training image collections.

2. RELATED WORK

The first problem to be taken into account is vision-based positioning. Different from the simultaneous localization and mapping (SLAM) in robotics, we use the model-based localization technic in this application. This localization technic consists of two main steps. Firstly you need to reconstruct a 3D model of the scene from many images or video sequences. Then you choose your query image and match it with the 3D model to compute the position. In the training phase, the application will first collect a sufficient number of images to construct the 3D point set model of a specific scene or area by using a structure from motion (SfM) algorithm [1]. In the paper, they present a system for interactively browsing and exploring large unstructured collections of photographs of a scene using a 3D interface. The image-based modeling method in our application follows their work.

In the positioning phase, one main problem is how to match features efficiently and effectively. Sattler et al. [2] propose a direct 2D-to-3D matching method based on visual vocabulary quantization and a prioritized correspondence search method to speed up the feature matching process. This method is extended to both 2D-to-3D and 3D-to-2D search for more efficiency [8].

In addition to positioning, compression methods have been developed to reduce the computational and memory requirements for large-scale models. Chen et al. [9] formulate the model compression algorithm as a weighted k-cover problem to obtain an optimal 3D point subset. They also combine the image-based modeling and the image-based positioning in their work, to achieve sub-meter positioning accuracy for Internet of Vehicle (IoV). Our application is based on their work of image-based positioning and we expands it to image retrieval.

3. SYSTEM OVERVIEW

As discussed in Section 1, the application consists of three phases. Figure 2 shows the system overview. The training phase is carried out on server or cloud systems. This phase runs when the image collections are ready. It performs image-based modeling to build a 3D point cloud model from collected images automatically. After the 3D model is built, the model compression will be performed to keep a small size.

The positioning phase is carried out on clients. When you take a picture or find a photograph of a scene that has been pre-modeled, you can send it to our application for the positioning phase. In this phase, this application performs feature-based 2D-3D matching on the local client. After matching the feature points of the image with the 3D point cloud model, the position where your picture is taken will be figured out.

The last phase, the retrieval phase is also carried out on the clients. The application uses the positioning result to query the list that contains the positions of all the images in the image collection, finding out the images that are closest to the query image and displaying them in the user interface. If the image collections have been annotated beforehand or there is GPS information along with the images, the application will also report the real position where the query image is taken on the interface.

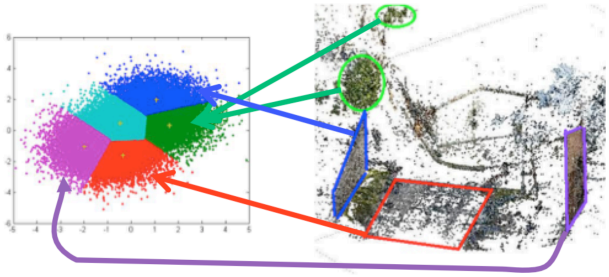

Fig. 3: An example of visual words clustering.

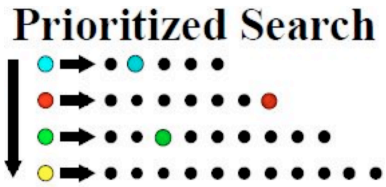

Fig. 4: An example of the prioritized search. Each color in the left represents a visual word. In the right are the candidate points. The searching order is from top to bottom.

4. POSITIONING-BASED IMAGE RETRIEVAL

As explained in Section 3, there are three main components in our system, which are described in the following sections.

4.1. Image-Based Modeling

Image-based modeling aims to construct a 3D point cloud model from a number of input images. One of the well-known image-based modeling systems is the Photo Tourism method [1], which uses structure from motion to estimate camera poses as well as reconstruct 3D scene geometry from images simultaneously. In this phase, we use a visual structure from motion implementation [10] method to build the 3D point cloud model. It has visual interface to use and also can be used by scripts.

Firstly, SIFT [11] features are extracted upon each image in the image collection. To reduce the computational time, a GPU implementation of SIFT [14] is used in our system. For every image pair, the feature point descriptors are matched with approximate nearest neighbors [12], and the fundamental matrices of all pairs are estimated using RANSAC, subsequently. In the next step, an incremental SfM method is used to avoid bad local minimal solutions and to reduce the computational load. It recovers camera parameters and 3D locations of feature points by minimizing the sum of distances between the projections of 3D feature points and their corresponding image features based on the following objective function:

$$\min_{c_j, P_i} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d\left(Q\left(c_j, P_i\right), p_{ij}\right), \qquad (1)$$

where c_j is the camera parameters of image j; m is the number of images; P_i is 3D coordinates of feature point i; n is the number of feature points; v_{ij} denotes the binary variables that equals 1 if point i is visible in image j and 0 otherwise; $Q(c_j, P_i)$ projects the 3D point i onto the im- age j; p_{ij} is the corresponding image feature of i on j; and $d(.)$ is the distance function. This objective function can be solved by using bundle adjustment, and a 3D point cloud model is built simultaneously. The model includes the positions of 3D feature points and the corresponding SIFT feature descriptor list for each point.

Because the 3D point cloud model is constructed in advance, a visual word tree or a visual vocabulary can be built before next step to speed up the image matching procedure. A visual word is a cluster of similar feature descriptors. An example of the clustering process is shown in Fig. 3. We use visual words instead of the original SIFT descriptors in the searching so that we can reduce the computing complexity. The visual words are obtained by k-means clustering. Afterwards, a kd-tree [15] is built based on the FLANN library [16] for fast indexing [2].

In the end of this phase, we store the reconstructed 3D point cloud model including the position and descriptor of each point. We also record the position of each training image and store them in a list to for the backward retrieval in the last retrieval phase.

4.2. 2D-to-3D Image Matching and Localization

2D-to-3D Image Matching and Localization aims to find the correspondences between 2D and 3D feature points and then compute the position of the 2D image in the 3D model. If we already have the 3D point cloud model, once there comes a query image, we first detect its SIFT features and compute the descriptors at the same time. Here we also uses the GPU implementation of SIFT to reduce the computational time. Then we perform 2D-to-3D matching to find the correspondence of the 2D points in the query image and the 3D points in the pre-built 3D point cloud model. The correspondence can be used to estimate the position.

In the positioning phase, we follow Sattler et al.'s method [2] to speed up the process. In their work, a prioritized correspondence search is applied. By associating 3D points to visual words, we can quickly identify possible correspondences for 2D features which are verified in a more detailed linear search. Here the 'prioritized' means we ensure the feature matching start with least candidate points in ascending order as shown in Fig. 4. The searching priority is from the visual word with the least candidate points to those with more candidate points. For example, if there are totally two visual words, 50 candidate points for visual word 1 and 100 candidate points for visual word 2, we will start feature matching with the visual word 1 for less computation and higher distinctiveness. The feature matching will stop when 100 correspondences have

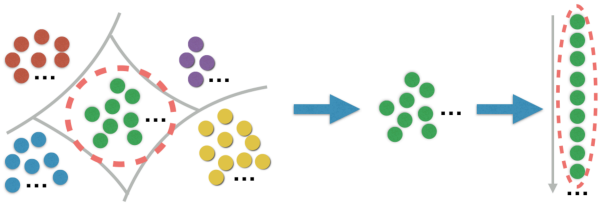

Fig. 5: The k-means 2D clustering used in the retrieval phase. To accelerate the search, we find the cluster first and then choose the nine training images that are closet to the query image in this cluster as shown above.

been found. The final 2D-to-3D correspondences are then used to estimate the position of the query image using the 6-point DLT algorithm [4] with RANSAC.

4.3. Image Retrieval Using Position Information

Having known the position of the query image, we can easily compute the distance between the query image and each training image. This is very simple, as we have stored the position of each training image in the training phase. By sorting the distances between the query image and each training image, we finally get the retrieval result. In our application, we set a threshold of 9 of all the retrieved images to find the best nine images matching with the query image.

In this retrieval phase, to reduce the computational time on clients when there are large numbers of training images, we need to optimize the searching way instead of using a simple exhaustive search. For this purpose, we propose a efficient search method by applying 2D clustering to the training dataset, which uses the k-means algorithm. This is an idea similar with the visual vocabulary used in the positioning phase that uses some clusters to replace the complete features. As a result, the unlabeled training images are classified into several clusters, which can be different areas in a scene where the training images are taken. When given the position of a query image, it will first determine which cluster this image or position belongs to, instead of performing the exhaustive search directly. It saves a lot of time and returns a few candidate images and their positions. Then we only need to compute the distances between the query image and these few candidate images to find the nine closest and best matching images. The distance between two images is computed by using the Euclidean distance:

$$\mathrm{dist}(\vec{x}, \vec{y}) = ||\vec{x} - \vec{y}||_2^2, \qquad (2)$$

where x and y are the positions of image 1 and image 2 in 2D coordinate. The whole flow of this search method is shown in Fig. 5.

In the other situation, where the training images are all manually annotated or have GPS information along with them, the clustering process can be simplified to manually dividing the scene into several parts or areas. When given the position of a query image, we first determine which part or area it belongs to by using some thresholds we set before, and then compute the Euclidean distances between the query image and all the candidate images in this part, as the last step in the former situation.

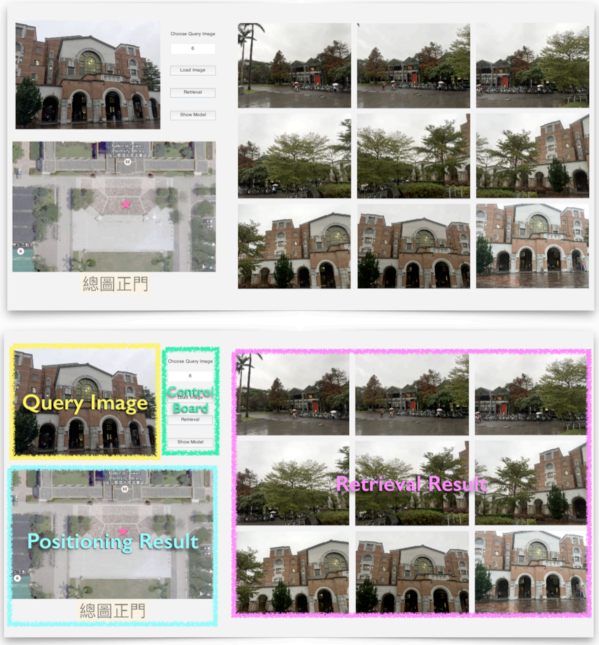

Fig. 6: The graphic user interface and the layout of our application. There are four main modules as shown in the lower half.

5. USER INTERFACE

This positioning-based image retrieval application has a graphical user interface which is easy to use. Our user interface is designed and implemented based on Matlab GUI, as shown in upper half of Fig. 6.

The lower half of Fig. 6 shows the layout of the interface. In the middle is the control board for some functions. You can enter a number and click the 'Load Image' button to choose and load a query image. The upper left part will show the query image you choose. After loading the query image you can click the 'Retrieval' button and the application will return nine best matching images on the right part of the interface. Furthermore, if we have annotated the training images before or have recorded the GPS information associated with the images, we can know the position in the real world and see the positioning result shown on the lower left part of the application. This positioning result will be shown as a marker projected on the vertical view of this scene associated with the text to describe the position.

There is another button for you to take a look into the point cloud called 'Show Model'. By clicking this button, the 3D viewer will open and show you the 3D

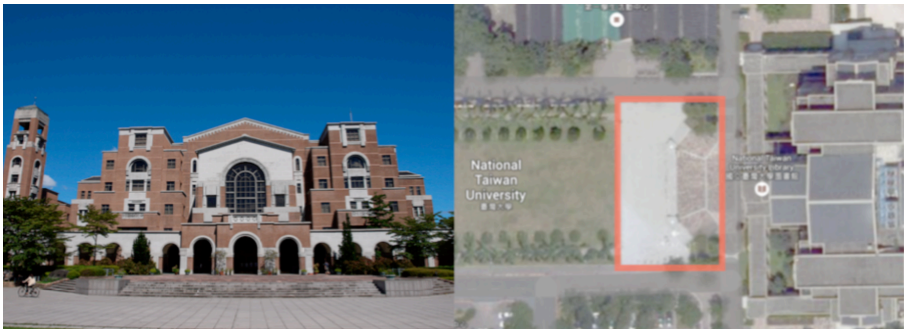

Fig. 7: Experiment scene where the training images and the query image are taken.

(a) (b)

Fig. 8: (a) Examples of the images collected from the experiment scene (i.e. the square in front of the NTU Main Library) at different positions and in different directions. (b) The reconstructed 3D point cloud model of the experiment scene (i.e. the square in front of the NTU Main Library). The markers are the positions of the training images.

point cloud model as well as the positioning result as a red marker in the model. You can use different operations to check the model, such as zooming, zooming out or 3D rotating. When using this interface for retrieval, the 3D point cloud model of the test scene needs to be built and linked as the necessary step before retrieval.

6. EXPERIMENTAL RESULTS

To verify the full functions of our application, we choose the square in front of the NTU Main Library for experiments, as shown in Fig. 7. Firstly we collect many pictures in this area, totally 369 images. These images are taken at different positions and in different directions as shown in part (a) of Fig. 8.

After collecting the images of the scene, we use the method [10] introduced before to build the 3D point cloud model as well as compute and record the position of each image simultaneously. In the experiment, we choose to manually annotate these images with the text description of the position. Then we store the model on the server. The pre-built model of the NTU Main Library is shown in part (b) of Fig. 8.

For testing, we collect 20 more images in this scene at different positions. When we want to use one of them for retrieval, we choose the number of the query image and click the 'Load Image' button. If the interface shows the correct image that we want to use, we can click the 'Retrieval' button to perform the positioning and retrieving. The 20 images are collected from five different areas in this scene, 4 testing images for each area. In our experiments, we use F-score to evaluate the retrieval accuracy. Widely used in image retrieval, the traditional F-measure or balanced F-score (F1 score) is the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \qquad (3)$$

The result of our experiment is shown in Table 1. In our experiment, we test the 20 testing images one by one,

Table 1: Retrieval accuracy in the experiments. There are 5 areas in the scene, with 4 testing images for each.

Area	#1	#2	#3	#4	#5
Images	4	4	4	4	4
Precision	36/36	36/36	36/36	36/36	36/36
Recall	36/36	36/36	36/36	36/36	36/36
F1	1.00	1.00	1.00	1.00	1.00

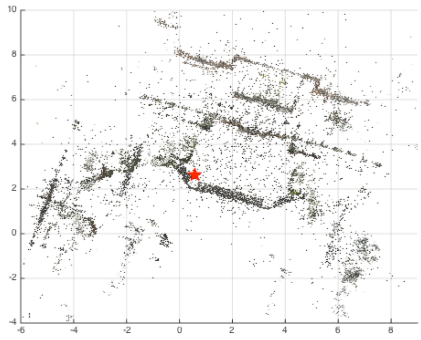

Fig. 9: The point cloud model shown in the 3D viewer after clicking the 'Show Model' button. The red marker is the position of the query image.

recording the precision and recall to compute F1 score for each testing.

As show in Table 1, the retrieval accuracy (precision and recall) in our experiment is 100%. It is because all the images are taken clearly and with sufficient features, so that the positioning results are accurate. Another reason is that the positioning error is small relatively and will not lead to obvious error in the retrieval result. The scale of the scene in the experiment is large enough compared with the small error. However, this retrieval method may still fail in some situations. The most possible one to happen is that the query image does not have enough features, such as the images of the white wall or the sky. Another situation is that the content of the image does not exist in any of the training images. In another word, the model is incomplete to cover the whole scene. So in this situation, we cannot match the query image with any 3D points the pre-built point cloud model to know the position.

At last we can check the model by using the 'Show Model' button. The model shown in the 3D viewer after clicking the button is shown in Fig. 9. The red star marker is your position in the scene.

7. CONCLUSION

In this paper we present an application to use positioning information for image retrieval and design a graphical use interface for it. For a scene where we already have many photos, we can use these photos to reconstruct the 3D model and store it for later use. When some one takes or gets a new photo of this scene, he can upload it to the application to match it with the model. The application will return his position, as well as other photos taken in the same position but maybe with different directions. With the help of our proposed search method in the retrieval phase, the processing time for each query image can be significantly reduced, especially when the training image collection is large. This retrieval method can maintain high accuracy as long as the images are taken clearly and have enough correspondence with others.

REFERENCES

[1] Snavely, Noah, Steven M. Seitz, and Richard Szeliski. "Photo tourism: exploring photo collections in 3D." ACM transactions on graphics (TOG). Vol. 25. No. 3. ACM, 2006.

[2] Sattler, Torsten, Bastian Leibe, and Leif Kobbelt. "Fast image-based localization using direct 2d-to-3d matching." Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE, 2011.

[3] N. S. Chang and K. S. Fu, A Relational Database System for Images, Technical Report TR-EE 79-28, Purdue University, May 1979.

[4] S.-K. Chang, C. W. Yan, D. C. Dimitroff, and T. Arndt, An intelligent image database system, IEEE Trans. Software Eng. 14(5), 1988.

[5] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, W. Equitz, Efficient and effective querying by image content, J. Intell. Inf. Syst. 3 (3–4) (1994) 231–262.

[6] A. Pentland, R.W. Picard, S. Scaroff, Photobook: content-based manipulation for image databases, Int. J. Comput. Vision 18 (3) (1996) 233–254.

[7] J.R. Smith, S.F. Chang, VisualSeek: a fully automatic content- based query system, Proceedings of the Fourth ACM International Conference on Multimedia, 1996, pp. 87–98.

[8] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. ECCV, 2012.

[9] Kuan-Wen Chen, Chun-Hsin Wang, Xiao Wei, Qiao Liang, Chu-Song Chen, Ming-Hsuan Yang, and Yi-Ping Hung, "Vision-Based Positioning for Internet-of-Vehicles," IEEE Transactions on Intelligent Transportation Systems, 2016.

[10] Wu, Changchang. "VisualSFM: A visual structure from motion system." (2011).

[11] D. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60(2):91--110, 2004.

[12] M. Muja and D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. VISAPP, 2009.

[13] R. Hartley and A. Zisserman. Multiple view geometry in computer vision. Cambridge University Press, 2004.

[14] C. Wu. SiftGPU: A GPU implementation of scale invaraint feature transform (SIFT). , 2007.

[15] D. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60(2):91--110, 2004.

[16] M. Muja and D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. VISAPP, 2009.