ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 9 (2019) No. 3, pp. 104-122
DOI:10.33168/JSMS.2019.0307

3D Reconstruction Using SfM with Sequence Images in Hadoop

Dongyue Wang & Taegkeun Whangbo

Department of Computer Science, Gachon University, Korean
[1] *wangdongyue89@126.com*

Abstract. Reconstructing a 3D structure from 2D images is a hot research topic in computer vision, and many approaches have been proposed to solve this problem in recent years. The structure from motion (SfM) algorithm has the potential to successfully reconstruct geometry from an image set, but the execution of the SfM algorithm on a single computer is only possible with small image sets. If an image set is large, the reconstruction speed decreases and it may even be too slow to complete 3D reconstruction. With the advent of cloud computing platforms, increasingly complex algorithms are being applied via cloud computing. Thus, the algorithm calculation speed for 3D reconstruction may be improved by taking advantage of the computational power of computer clusters. In this paper, we propose an algorithm for realizing 3D reconstruction from sequence images on the Hadoop cluster. Our test results reveal that, using Hadoop, we can achieve fast 3D reconstruction.

Keywords: 3D reconstruction, Structure from motion, Hadoop, HIPI, Bundler adjustment.

1. Introduction

Computer vision is a simulation of human vision. The real world is three-dimensional (3D); however, the images obtained using a camera are two-dimensional (2D). Therefore, obtaining a three-dimensional model from an image is a hot topic in computer vision. Inspired by how the human eyes obtain information, researchers have proposed a multi-view-based 3D reconstruction algorithm (Hartley and Andrew 2003). Image-based 3D reconstruction generally refers to obtaining the 2D image information of a scene or object at different perspectives. First an image is obtained using a camera. Then, the image is analyzed and processed to restore the spatial geometry of the original 3D scene

or object. Compared to traditional geometry-based modeling, multi-view 3D reconstruction is a low cost, user-friendly operation, and a strong sense of reality. In recent years, multi-view 3D reconstruction has become a hot research topic in the fields of computer vision and computer graphics. Researchers have also proposed a large number of multi-view-based 3D reconstruction algorithms, such as contour-based methods, texture-based methods, and structure from motion (SfM) (Wang and Whangbo 2019; Snavely 2008). In this paper, we use SfM for 3D reconstruction. In SfM, reconstruction is achieved by the following sequence of steps. First, the feature points from the image sequence are detected. Then, the feature points are matched. Subsequently, the parameters of the camera model are calculated using the multi-view geometric constraint relationship. Finally, the camera parameters are reconstructed to form the 3D model of the scene. The advantage of SfM is that the reconstruction process depends only on the feature point matching between the views and no additional equipment is needed. For this reason, many researchers are using SfM for 3D reconstruction. However, with an increasing number of images, huge amounts of time are required to realize 3D reconstruction on a single computer, meaning requests for fast reconstructions cannot be fulfilled. Numerous research efforts have been made to boost computer performance and utilize GPUs to extract feature points to improve calculation speed. However, the improvements realized by these methods are marginal at best because of the complexity of the SfM algorithm.

The powerful computing abilities of the Hadoop computing platform (Chang, et al., 2008) have made it a popular tool for researchers. A number of complex algorithms have been successfully implemented on Hadoop for performing calculations. We were motivated to try to improve the algorithm calculation speed for 3D reconstruction using the computing power of the Hadoop computer cluster, which serves as a platform in which a number of computers can perform computations concurrently, resulting in improved efficiency. In this study, we implemented the SfM 3D reconstruction algorithm on the Hadoop cluster and confirmed that its utilization can contribute to fast 3D reconstruction.

In section 2, we review related works on SfM and Hadoop. In section 3, we describe the implementation of the SfM 3D reconstruction algorithm on Hadoop to improve feature matching and calculation speed. We describe our experimental process and evaluate the reconstruction results in section 4. Finally, in section 5 we present our conclusions and discuss future plans.

2. Related Work

2.1. Structure fron motion

The SfM algorithm, which was proposed by Longuet-Higgins, has been a

research hotspot in various fields since its inception. Early SfM technology was based on photogrammetry, which required camera calibration during reconstruction to obtain the internal parameters of the camera and confirming camera position and direction. A 3D structure can only be reconstructed when the internal and external parameters are equal. In order to achieve more accurate results, bundle adjustment optimization is typically performed on the initial reconstruction results. Bundle adjustment has become an important part of SfM. SfM methods can be divided into two classes: sequential and global. Sequential SfM pipelines start with a minimal reconstruction based on two or three views and incrementally add new views into a merged representation. This class of SfM algorithm uses the following procedure:

- Feature detection and matching: the first step is feature point extraction and matching. Feature point extraction is the basis of 3D reconstruction. We must calculate the positional relationship of the camera by matching feature points. Therefore, the effect of feature point matching directly determines the success or failure of 3D reconstruction. Common feature point extraction algorithms include SIFT, SURF, and Harris. After extracting the feature points, we must match the feature points.
- Computation of the fundamental matrix and essential matrix: The second step is the computation of the fundamental matrix and essential matrix. After feature point matching, we must calculate the fundamental matrix according to the matched feature points. The basic matrix is a 3 x 3 matrix. This matrix represents the correspondence between two matching feature points. Then, the camera internal parameters are used to solve the essential matrix. The camera internal parameters are the values obtained by camera calibration. The essential matrix contains the relative rotation of the camera and the matrix of the translation. After obtaining the rotation matrix R and the translation matrix T by SVD decomposition to find the external parameters of the camera.
- Calculated camera position and 3D point: The third step is the calculation of 3D points based on camera position. Using the camera model, we can calculate the 3D points based on the principle of the pinhole camera. We then apply trigonometry to solve the 3D point.
- Bundle Adjustment: Finally, we use the Bundle Adjustment algorithm to optimize and output the result of 3D reconstruction.

(1) Feature detection and matching

In the SfM algorithm, the starting points of reconstruction are the extraction and matching of feature points. From among the common feature detection algorithms, including scale invariant feature transform (SIFT) (Lowe, 2004),

SURF, Harris, etc., we chose improved SIFT to ensure high accuracy of feature matching.

David Lowe first proposed the SIFT algorithm in 1999. It is still used to detect and describe local features in images. The SIFT algorithm is characterized by invariance to rotation, scale scaling, and brightness variation. Furthermore, it maintains a certain degree of stability against viewing angle changes, affine transformation, and noise. It has unique, multi-volume, high-speed, and scalable properties. Therefore, the SIFT algorithm can address issues, such as rotation, scaling, translation of the target, image affine transformation, projection transformation, illumination influence, target occlusion, debris scene, noise. The essence of the SIFT algorithm is to find key points (feature points) in different scale spaces and calculate the direction of the key points. The key points that SIFT finds are outstanding and do not change due to factors such as illumination, affine transformation and noise. For example, corner points, edge points, bright spots in dark areas, and dark spots in bright areas. Therefore, the SIFT algorithm is an effective feature point extraction algorithm. The SIFT algorithm is composed of four main stages of computation for generating a set of image features.

The first stage is scale-space extrema detection. In this stage, the algorithm searches for images in all scale spaces and identifies potential scales and invariable points of interest through Gaussian differential functions. To detect stable key points in scale space, Lowe proposed the detection of local extremum points in Gaussian difference scale space as the key point. The DoG operator is defined as the difference between the Gaussian kernels of two different scales. The DoG equation is as follows:

$$\mathrm{D}(\mathrm{x},\mathrm{y},\sigma) = \big(G(x,y,k\sigma) - G(x,y,\sigma)\big) * \mathrm{I}(\mathrm{x},\mathrm{y}) = L(x,y,k\sigma) - L(x,y,\sigma) \quad (1)$$

where G (x, y, σ) is a Gaussian kernel function, is a scale space factor, which is the standard deviation of the Gaussian distribution. It reflects the degree to which the image is blurred. The larger the value of σ, the more blurred the image, and the larger the corresponding scale becomes.

The second stage is key point localization. Given that the DoG value is sensitive to noise and edges, and its positioning accuracy is not high, to enhance the stability of the feature and improve the positioning accuracy, SIFT performs curve fitting through the scale space DoG function to remove feature points that do not meet the requirements. There are two main feature points that do not meet the requirements. The first is a low-contrast feature point, and the other is the corresponding point on the edge. Once these steps are complete, the feature points are detected. For the image to become rotation invariant, it is necessary to calculate the direction of the feature points. According to the gradient direction

distribution feature of the neighborhood pixel of the feature point, a direction can be specified for each feature point. This can make the operator rotation invariant. Let (x, y) be a neighboring pixel of a feature point, then the modulus of the gradient at (x, y) pixels, m(x, y), and the direction θ(x,y) are calculated as follows:

$$\mathrm{m}(\mathrm{x},\mathrm{y}) = \sqrt{[L(x+1,y)-L(x-1,y)]^2+[L(x,y+1)-L(x,y-1)]^2} \quad (2)$$

$$\theta(\mathrm{x},\mathrm{y}) = \arctan\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)} \quad (3)$$

The third step is orientation assignment. After calculating the gradient direction, the histogram is used to calculate the gradient direction and amplitude corresponding to the pixels in the neighborhood of the feature points. The gradient histogram changes from 0 to 360 degrees. At each 45 degrees is a column; thus, a total of eight columns exist between 0 and 360 degrees. The peak of the histogram indicates the main direction of the neighborhood gradient at the current feature point, which is the direction of the feature point.

The final stage is the key point description. First, the coordinate axis is rotated to be consistent with the main direction of the feature point such that rotation invariance is ensured. Then, a 16 x 16 window is selected, centering on the feature point, and then divided into 16 4 x 4 squares and calculated in the eight directions gradient histogram. Thus, each feature point can generate 128 data samples, which is the 128-dimensional feature vector of SIFT. After generating the SIFT feature vector, we can establish the correspondence between the feature vectors using a similarity measure such as the Euclidean distance between two feature points for matching (Matthew Brown, et al), using the SIFT algorithm for feature point detection and matching. The result is shown in Fig.1.

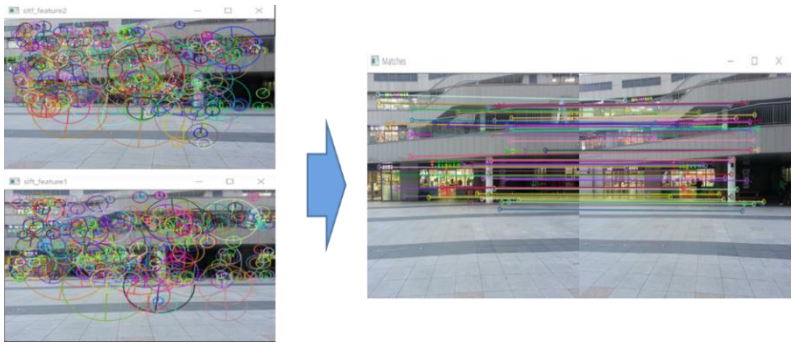

Fig. 1: Feature detection and matching results using SIFT

(2) Computation of the fundamental matrix and essential matrix

In the two images, the relative position of the matched feature points is

constrained by the geometric relationship of the two image planes, which is called the polar geometry in computer vision. The fundamental matrix is an algebraic representation of epipolar geometry (Dubey and Tomar, 2017). It is a 3×3 matrix of rank two. If a point in 3-space X is imaged as x in the first view and x' in the second view, then the image points satisfy the relationship:

$$x' F x = 0 \tag{4}$$

where x' and x are the corresponding feature points in the two images. At least eight corresponding points are required to calculate the fundamental matrix F, which is a 3×3 matrix of rank two.

The essential matrix E (Hartley, 1992) is a specialization of the fundamental matrix for the case of normalized image coordinates. Considering a pair of normalized camera matrices P=[I|0] and P'=[R|t], the fundamental matrix corresponding to the pair of normalized cameras is customarily called the essential matrix and has the form:

$$E = [t]_x R \tag{5}$$

In equation 5, t represents the relative translation of the two cameras and R is a rotation matrix. Thus, E contains the relative positions of the two cameras. The relationship between F and E is:

$$E = K'^T F K \tag{6}$$

In the equation 6, K' and K are the internal parameters of the two cameras. The camera internal parameters represent the projection relationship inside the ray machine. We usually use the camera calibration to find the camera internal parameters. Once we know the internal parameters of the two cameras, the essential matrix E can be calculated using equation 6.

(3) Calculated camera position and 3d point

The essential matrix includes the relative positions of the two cameras, which can be obtained by decomposing the essential matrix (Triggs, 1996). The essential matrix has a rank of two, which means exactly two of its singular values are non-zero. In contrast to the fundamental matrix, the essential matrix satisfies the additional constraint that these two singular values are equal because of the invariant singular values of a matrix after the orthonormal transformation of that matrix. Thus, the singular value decomposition of E is:

$$E = U D V^T \tag{7}$$

where D=diag (k, k,0). Then, up to a certain scale factor, the factorization has one of the forms:

$$S = V Z V^T \quad R = U W V^T \text{ or } U W^T V^T \tag{8}$$

Where:

$$W = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad Z = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{9}$$

In equation 8, R and S are the rotation and translation matrices, respectively. The origin of the world coordinate system can be set as the optical center of the camera of the first image, meaning the camera projection matrix for the first image can be expressed as:

$$P = K\,[\,I \mid 0\,] \tag{10}$$

Where I is a 3×3 identity matrix. Then, R is the rotation matrix and t is a translation vector for the camera of the second image. The camera projection matrix for this camera can be represented as:

$$P2 = K\,[\,R \mid t\,] \tag{11}$$

After the camera model is obtained, the 3D coordinates of the corresponding feature points can be computed via triangulation. The triangulation model is shown in Fig. 2.

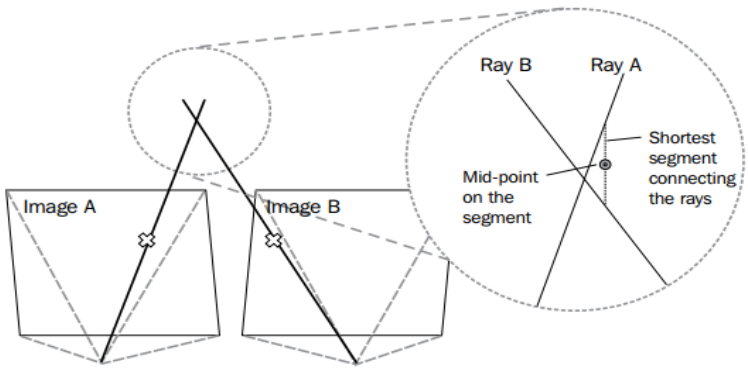

Fig. 2: Feature detection and matching results using SIFT

In Fig. 2, images A and B are images taken by cameras at different positions. After finding the corresponding points on the image, we can derive the 3D coordinates of the feature points based on the camera model. In the ideal case, rays A and B should intersect. However, because the two lines do not intersect because of errors, we typically use the center point with the shortest distance between the two rays as the 3D coordinate to reduce the error.

(4) Bundle adjustment

After calculating the 3D point, we use the bundle adjustment (S. Agarwal et al, 2010) method to reduce error. The bundle adjustment method takes the image

point coordinates of undetermined pixels as observations and uses minimum reprojection error as the objective function to solve for the camera external parameters and space coordinates of undetermined pixels using the least squares principle. It belongs to the class of nonlinear minimization algorithms. The bundle adjustment process can be expressed as:

$$g(C, X) = \sum_{i=1}^{n} \sum_{j=1}^{m} \omega_{ij} (q_{ij} - P(C_i, X_j)^2 \quad (12)$$

where $P(C_i, X_j)$ is the reprojection value and q_ij is the observed 2D location of point j in image i. The goal of bundle adjustment is to minimize the value of the objective function g (C, X).

2.2. Hadoop

Apache Hadoop is an open-source software framework for distributed storage and distributed processing of very large data sets on computer clusters built from consumer hardware. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Hadoop consists of two important frameworks: The Hadoop distributed file system (HDFS) (Ghazi, et al) and mapreduce. Featuring high fault tolerance and scalability, HDFS allows users to deploy Hadoop on cheap hardware for constructing distributed systems. HDFS supports streaming forms to access data. MapReduce is applied to massive data analysis problems. Combining MapReduce and HDFS, the big data can first be divided, and then parallel calculation can be performed on the divided data. The output of the previous stage is used as the input of the next stage. Thus, the data can be processed efficiently in the distributed cluster. The distributed computing framework of MapReduce allows users to develop applications without any special knowledge regarding distributed systems. Distributed applications utilize large-scale computing resources to solve big-data problem that cannot be solved using traditional computers.

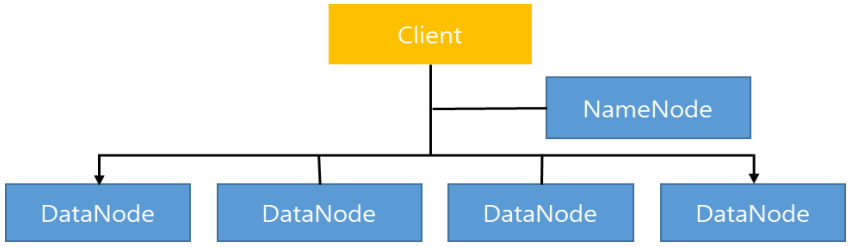

Fig. 3: The structure of Hadoop system

(1) Mapreduce

MapReduce adopts a structure similar to that of HDFS and includes

JobTracker and TaskTracker. The function of JobTracker is to manage the jobs in Hadoop. The function of TaskTracker is to perform specific job tasks. The core processes of MapReduce is mapping and reducing. The map process processes the data block and outputs the result in the form of key value pairs. The system sorts the results according to the map key as an input to the reduce process. A MapReduce job is divided into the following steps:

- Submit homework.
- Assign and execute Map tasks.
- Assign and execute reduce tasks.
- Output the result.

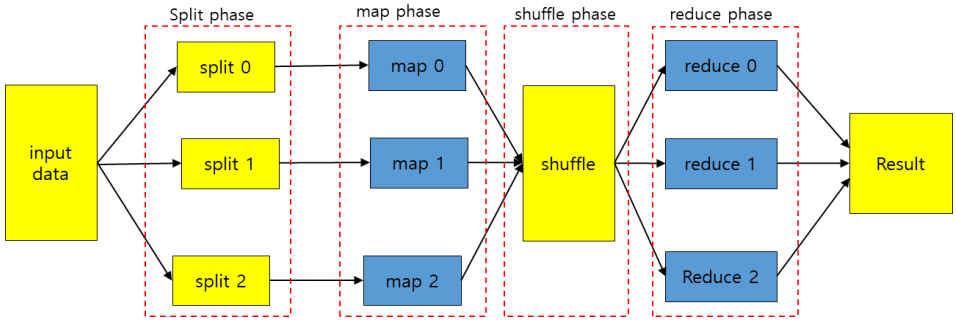

Fig. 4: The mapreduce structure

(2) Hadoop image processing interface

In order to process images on Hadoop, the University of Virginia Computer Graphics Lab proposed the Hadoop image processing interface (HIPI) (Sweeney and Arietta, 2011). The HIPI is an image processing library designed to be used with the Apache Hadoop MapReduce parallel programming framework. It provides a solution for storing large collections of images on the HDFS and makes them available for efficient distributed processing. The HIPI also provides integration with OpenCV, which is a popular open-source library that contains many computer vision algorithms. The organization of a typical HIPI program is illustrated in Fig. 5.

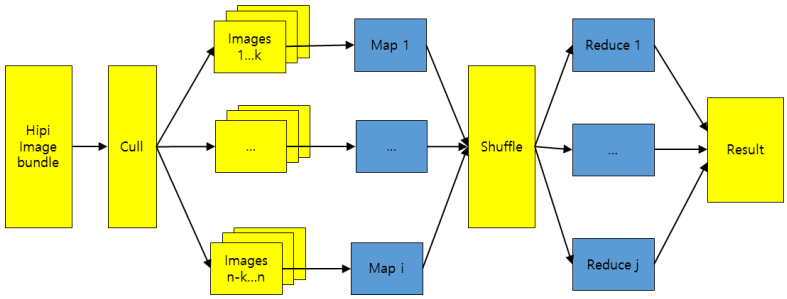

Fig. 5 : Organization of a typical HIPI program

3. Proposed Algorithm

As described in section 2.1, the SfM computing process is complex and requires a huge amount of time to perform 3D reconstruction when there are many images. For example, approximately 10 minutes are required for the 3D reconstruction of 100 images on a single computer. Several days may be required for 3D reconstruction when using large-scale image sets (e.g. numbers of images greater than 10,000). This makes it impossible to satisfy requests for fast reconstruction.

With the advent of the Hadoop computing platform, a growing number of complex algorithms (Yan and Huang, 2014; Chandrika, 2018) have been applied to the Hadoop platform to achieve reduced run time. In this study, in order to improve the efficiency of 3D reconstruction, we implemented a 3D point cloud reconstruction algorithm using SfM with sequence images in Hadoop.

Because Hadoop is a distributed system composed of mapper and reducer phases (He, et al, 2008) the following changes are required for the implementation of our algorithm:

- First, image formats must be converted into those that can be identified by Hadoop. In this study, we converted the images using the HIPI image bundle because we used the HIPI library.
- In the mapper phase, we performed feature extraction using the improved SIFT algorithm and changed to the key value format.
- In the reducer phase, after matching the extracted feature points, we calculate the fundamental matrix F and essential matrix E by matching pairs of feature points. The then obtained the relative camera positions by factorizing the essential matrix. Finally, we utilized the camera position information to calculate 3D points. Because errors occur during the reconstruction of 3D points because of noise, we performed bundle adjustment to for error reduction. Fig. 6 presents a diagram of the system composition.

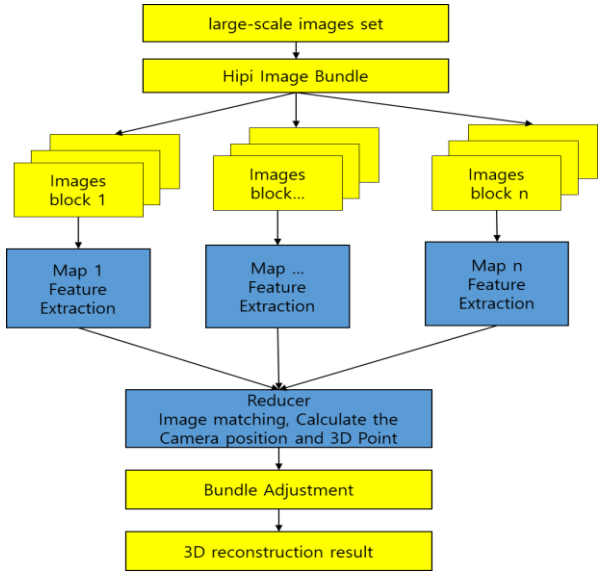

Fig. 6: System composition diagram

3.1. Image imput and conerdion format

Hadoop was designed for traditional large-text-file processing utilities in which each line is a record. Because images are not text files, they cannot be directly analyzed by Hadoop. Therefore, we first converted the image files into a format identifiable by Hadoop in order to make them compatible with the platform. We used the HibImport format conversion tool from the HIPI library. From a folder of images on a user's local file system, HibImport creates a Hipi Image Bundle (HIB) as the main input file for the HIPI framework. A HIPI Image Bundle consists of two files: a data file containing concatenated images and an index file containing information regarding the offsets of images in the data file. This setup allows for easy access to images anywhere in the bundle without having to read every image.

3.2. Mapper phase

In the mapper phase, the first step is to get the Hib images and then convert them to the Mat image format. Given that the extraction of the feature points is a key part of the 3D reconstruction, and is thus directly related to the success or failure of the 3D reconstruction, we apply the SIFT algorithm to detect the feature points of an image in this study. Which causes each feature point to generate a 128 dimensional feature point descriptor. Since all data stores in the Hadoop file system are in key value format, we need to store the feature point descriptors in the form of key values. The value of key takes the name of the

image plus the index value of the feature point, and the value of value includes the position of the feature point, the RGB value of the feature point, and the descriptor of the feature point. The format of the key value of the extracted feature points is as follows:

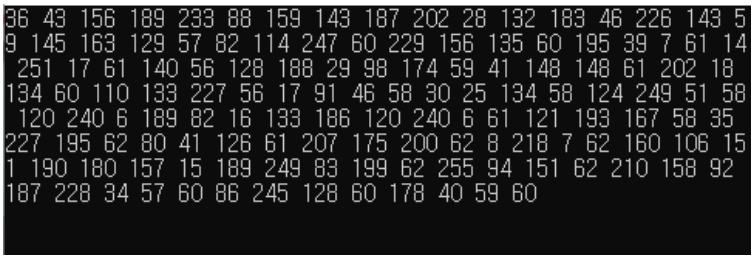

Fig. 7: Convert the feature point descriptor to the output value of the key value. The first two digits represent the position of the feature points, the next three digits represent the RGB values of the feature points, the last digit is the description vector of the feature points, and each feature point has a 128-dimensional description vector.

The final mapping output is a key-value pair. Next we will enter the Reducer phase and perform feature point matching.

3.3. Reducer phase

In the reducer phase, the feature points of the image must be matched after extracting the feature points of all the images in the mapper stage. Given that the sequence image is used, we only need to match (Kim and Manjusha, 2017) the two adjacent images. When matching, we use the RANSAC algorithm to improve the matching accuracy. The matching of adjacent images is in fig. 8:

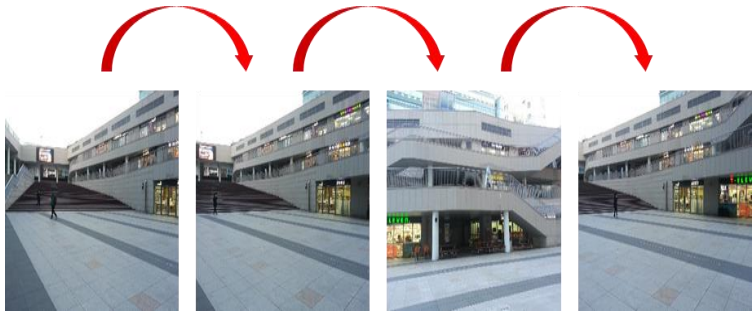

Fig. 8: Matching of two sequence images

After the feature points matching, we use the matched feature points to calculate fundamental matrix and essential matrix, Because the Essential matrix

contains camera external parameters, we use SVD decomposition to calculate the camera model. and use factorization to obtain an initial camera model. We then use triangulation to obtain the 3D points that form the 3D reconstruction of two images.

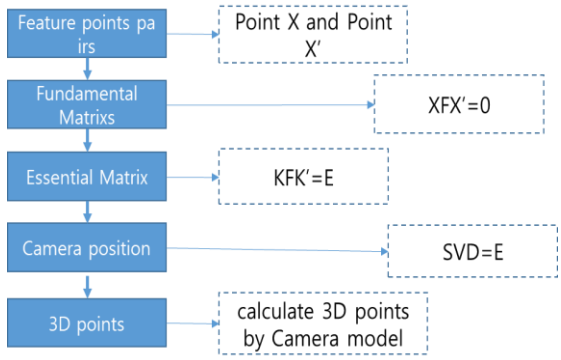

Fig. 9: Calculation process in the reduce phase

After reconstructing the feature points of the two images, we'll add more 3D points in the image, we use a sequential method to perform 3D reconstruction for more than two images. Sequential SfM pipelines begin with a minimal reconstruction based on two views. New views are then incrementally added into a merged representation. The sequential SfM algorithm follows the steps shown in in figure 10.

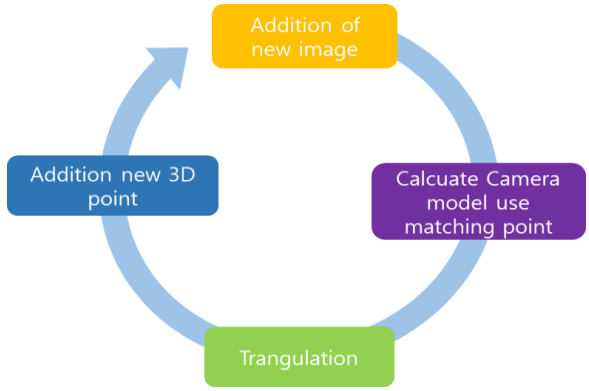

Fig. 10: Sequential SfM algorithm for 3D reconstruction

Figure 8 illustrates the process of point cloud integration. In the beginning, we utilize the initial two images to the first point clouds. The rest of the images

are then reconstructed one by one, followed by adding the new point from every image into the point cloud until all of the images are completely reconstructed.

Because errors occur in the reconstructed 3D points because of noise. We use the bundle adjustment method to reduce the reconstruction error. The bundle adjustment method optimizes the back projection error between the actual detected and predicted image points, which can be expressed as the sum of the squares of the nonlinear real-valued functions, so the minimum can be obtained by nonlinear least squares. We use the Levenberg-Marquardt algorithm to minimize the error.

4. Experimental Results and Evaluation

To evaluate the performance of the proposed algorithm, we implemented it on a single computer and on the Hadoop platform using the OpenCV library. The Hadoop platform is a Hadoop cluster composed of five computers. We utilized the following libraries in this experiment:

- Linux version Ubuntu 16.04
- OpenCV version 2.4.11
- Hadoop version 2.7.2
- HIPI version 2.1.0

Because the SfM algorithm output is data based on 3D point cloud and camera positions, to visualize the reconstruction results, we displayed the reconstructed 3D point cloud using the Point Cloud Library (PCL) (Rusu and Cousins, 2011).

First, we used 100 standard database photos taken by a single calibrated camera to perform 3D reconstruction. These photos were taken by Steve Seitz, James Diebel, Daniel Scharstein, Brian Curless, and Rick Szeliski Fig. 11 and 12 present the input images and results, respectively.

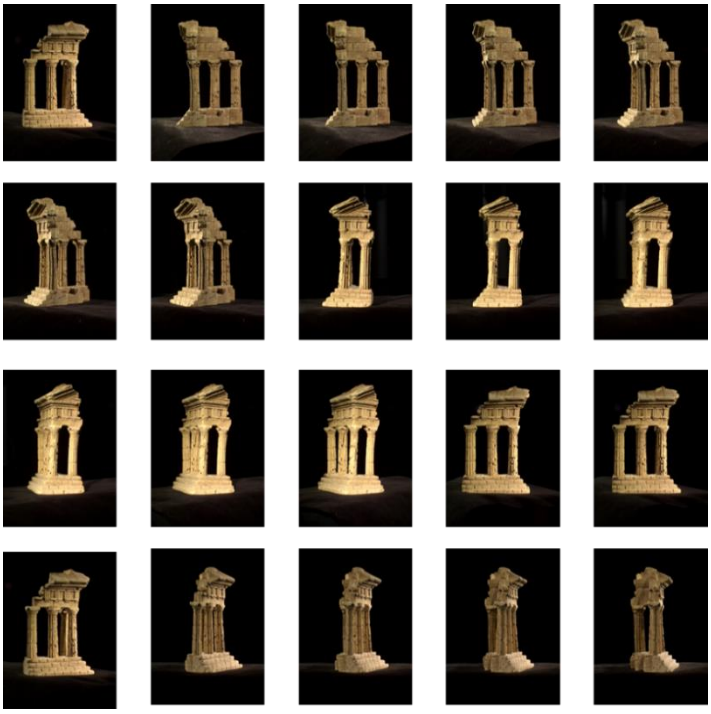

Fig. 11: We used 100 input sequence images photographed from different positions to perform 3D reconstruction, 20 of which are illustrated here.

Fig. 12: 3D reconstruction results using 100 input sequence images. The yellow area represents the 3D reconstructed points and the red points indicate the camera positions. Some red points overlapped because they were drawn large and tightly clustered.

Next, we used 100 sequence images captured by a smartphone camera depicting the vision tower at Gachon University for 3D reconstruction. Figs. 13 and 14 present the input images and results, respectively.

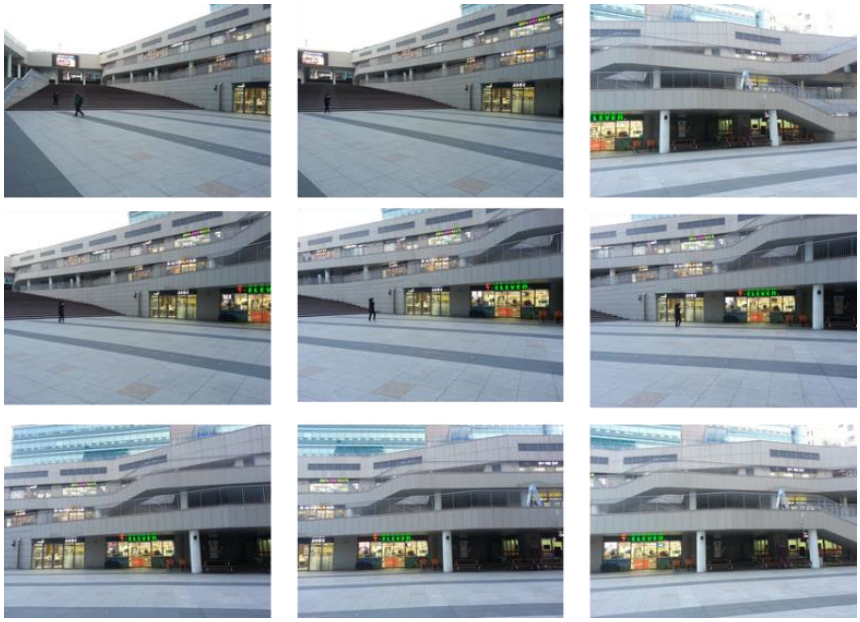

Fig. 13: Input images depicting the vision tower at Gachon University.

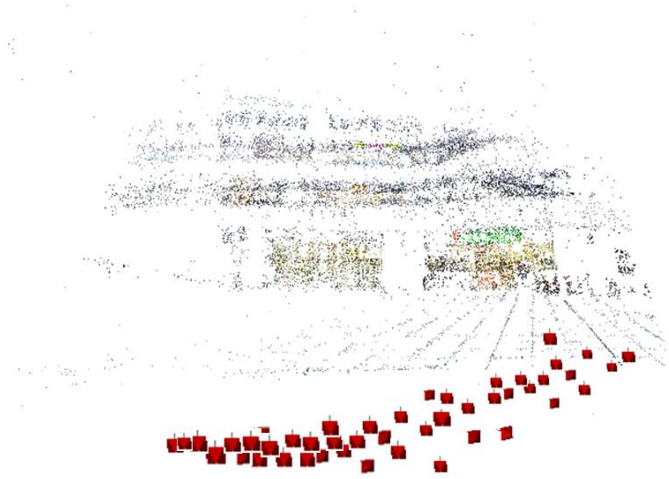

Fig. 14: 3D reconstruction results using sequence images depicting the vision tower at Gachon University.

For comparison, we performed the same 3D reconstruction on a single computer. Table 1 contains the elapsed times for the 3D reconstruction of these images.

Table 1: Comparison of time consumed for 3D reconstruction using 100 images

Title 1	Single computer	Hadoop
standard database photos	7 minutes and 43 seconds	3 minutes and 21 seconds
vision tower images	9minutes and 32 seconds	4 minutes and 39 seconds

As shown in Table 1, 3D feature points can be reconstructed on the Hadoop platform much faster than on a single computer.

5. Conclusion

Computer vision is an important research field and 3D reconstruction based on image sequences has significant research and application potential. Compared to traditional modeling methods, image-based 3D reconstruction has the advantages of being a simple and fast construction method, having no measurement requirements, and creating a strong sense of reality. It has garnered significant research interest in recent years. Using the SfM algorithm, we performed a number of experimental analysis tasks.

First, to achieve 3D reconstruction from image sequences, it is necessary to determine the matching relationships between the image feature points. We used SIFT features in the matching algorithm to improve the matching accuracy of feature points. Owing to its rotation invariance, scale invariance, and strong stability to illumination and occlusion, SIFT is effective in feature extraction. By using the SIFT matching algorithm, the accuracy of the feature point extraction is significantly improved, and more relevant information regarding the feature points is obtained. This is then used for the next calculation step.

For 3D reconstruction, we used the SfM algorithm to calculate the positional relationship of feature points between a series of sequence images, rotation, and so on. From the calculation of the fundamental matrix and the essential matrix, the SVD decomposition is performed to obtain the camera matrix, and R and t are obtained. Thereby, depth information between the images and 3D feature point information are calculated. Next to solve the issue of slow reconstruction or failure to complete reconstruction for large-scale image sets, we proposed implementing the 3D reconstruction algorithm on the Hadoop platform to enhance 3D reconstruction speed. Porting the 3D reconstruction method to Hadoop is a major innovation in our algorithm. Reconstruction by SFM algorithm has a good effect, but its calculation process requires a lot of time, which is a

problem in 3D reconstruction. Therefore, we propose to perform 3D reconstruction on the Hadoop platform. It turns out that our algorithm has greatly improved its speed while maintaining good reconstruction results.

Although our proposal to perform 3D reconstruction on Hadoop was successful in some respects, there are various areas in need of improvement. First, although we improved feature point matching, errors still occurred during the matching process. Second, our final 3D reconstructions failed to reconstruct the actual dimensions of target objects. Therefore, in future research, we will focus on the improved matching of feature points and realization of superior Euclidean reconstruction.

References

Bingsheng He; Wenbin Fang; Qiong Luo; Naga K. Govindaraju; TuYong Wang. Mars: a MapReduce framework on graphics processors. *Proceedings of the 17th International Conference on Architectures and Compilation Techniques*, (2008): 260-269.

Bill Triggs. Factorization Methods for Projective Structure and Motion, *Computer Vision & Pattern Recognition*, (1996):845-851.

Deepika Dubey, G. S. Tomar, "Echelon Based Pose Generalization of Facial Images Approaches", Asia-pacific Journal of Convergent Research Interchange, SoCoRI, ISSN : 2508-9080 (Print); 2671-5325 (Online), Vol.3, No.1, March (2017), pp. 63-75, http://dx.doi.org/10.21742/APJCRI.2017.03.06

D. G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints.” *International Journal of Computer Vision*, vol.60, no.2, (2004):91-110.

Dong Jo Kim, P. Lakshmi Manjusha, "Building Detection in High Resolution Remotely Sensed Images based on Automatic Histogram-Based Fuzzy C-Means Algorithm", *Asia-pacific Journal of Convergent Research Interchange, SoCoRI*, ISSN : 2508-9080 (Print); 2671-5325 (Online), Vol.3, No.1, March (2017), pp. 57-62, http://dx.doi.org/10.21742/APJCRI.2017.03.05

Dongyue Wang, Taegkeun Whangbo. “3D Point Cloud Reconstruction Using Structure from Motion with Sequence Images in Hadoop.” *International Journal of Image and Signal Systems Engineering*, Vol. 3, No. 1 (2019)

Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wa llach, Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gryber. Bigtable. “A distributed storage system for structured data. Proceedings of the 7th Symposium on Operating System Design and Implementation.” *WA*, (2006): 205-218.

G. Chandrika, "Study on Software Reliability and Reliability Testing", *Asia-pacific Journal of Convergent Research Interchange, SoCoRI*, ISSN : 2508-9080 (Print); 2671-5325 (Online), Vol.1, No.1, March (2015), pp. 7-20, http://dx.doi.org/10.21742/APJCRI.2015.03.02

Keith N. Snavely. "Scene Reconstruction and Visualization from Internet Photo Collections", Doctoral thesis, University of Washington, Seattle, *WA*,2008, pp.1-192.

Matthew Brown; Richard Szeliski; Simon Winder. Multi-Image Matching using Multi-Scale Oriented Patches, *In Computer Vision and Pattern Recognition*, 510-517.

Mohd Rehan Ghazi; Durgaprasad Gangodkar. Hadoop, MapReduce and HDFS: A Developers Perspective, *Procedia computer science*, volume 48, 45-50

L. L. Chris Sweeney; J. L. Sean Arietta. HIPI: A hadoop image processing interface for image-based map reduce tasks, University of Virginia, Department of Computer Science 2011.

QUAN-TUAN LUONG; OLIVIER D. FAUGERAS. The Fundamental Matrix: Theory, Algorithms, and Stability Analysis, *International Journal of Computer*, Volume 17, (1996): 43-76.

Radu Bogdan Rusu; Steve Cousins. 3D is here: Point Cloud Library(PCL), In IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 2011.

R. Hartley. Estimation of Relative Camera Positions for Uncalibrated, Cameras. Computer Vision, (1992): 579-587.

R. Hartley; Z.Andrew. Multiple View Geometry in Computer Vision, Cambridge University Press, 2003.

S. Agarwal; N. Snavely; S. Seitz; R. Szeliski. Bundle adjustment in the large, In ECCV10, (2010): 29–42.

Yuzhong Yan; Lei Huang. Large-Scale Image Processing Research Cloud, *Fifth international conference on cloud computing, GRIDs, and virtualization*, (2014): 83-93.