A MapReduce Approach for SIFT Feature Extraction

Wei Han*, Yiding Kang**, Yang Chen* and Xueqing Zhang*
* The 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang, China
** The Aeronautical University of China People Liberation Air Force, Changchun, China

Abstract—**SIFT feature extraction is a computationally intensive problem, for the large scale image, which will take a long time to extract SIFT feature. This paper presents a novel approach, based on MapReduce, to accelerate SIFT feature extraction. A MapReduce based SIFT feature extraction model is established, and the original SIFT feature extraction progress is reformed to fit the model. We have implemented the MapReduce based algorithm and evaluated it on a Hadoop cluster. The experimental results show that this approach can extract SIFT feature simultaneously on Hadoop cluster with a good speed up rate.**

Keywords- MapReduce; SIFT; feature extraction; big data; hadoop

I. INTRODUCTION

A local feature is an image pattern which distinguishes from its immediate neighborhood. It can effectively handle various challenges such as partial occlusion, illumination changes etc. As a result, it widely used in many fields, such as image stitching, object detection, image retrieval and etc. Since the nineties of the last century, a very large variety of local feature extraction algorithms have been proposed. These algorithms can be broadly divided into two categories: point based methods (such as SIFT[1], SURF[2], and Hessian-Laplace[3] etc.) and region based methods (such as MSER[4] etc.).

SIFT(scale invariant feature transform) local feature, proposed by Lowe in 1999, is the milestone work on local feature research. Mikolajczyk[5] compare the performance of a variety of representative algorithms under different scenarios, including illumination changes, Image geometric distortion, resolution difference, rotation, blur, image compression. And the results shows that SIFT presents the best performance.

But SIFT feature extraction is a computationally intensive problem. For the large scale image, therefore, it will take a long time to extract SIFT feature. There are large requirements of fast SIFT feature extraction algorithm in many fields. Parallel processing technology is an effective approach to improve the processing speed. Along with the expansion of the system, the traditional parallel system architecture increasingly complex. Load imbalances, programming, storage and energy consumption, and other issues all affect the efficiency of parallel computing. The larger the system, the higher communications cost it takes. The effectiveness of the algorithm is also likely to be affected by node failures. So parallel computing must use highly reliable servers and disks, making it a costly computing model.

MapReduce[6], invented by Google, is a distributed and parallel processing programming model for processing tasks on large data sets using distributed clusters of commodity computers. MapReduce simplifies parallel data processing methods on large clusters and shows very good scalability. Through a simple calculation model, MapReduce hides many very vexing problems in parallel and distributed computing such as data parallel processing, fault tolerance, data distribution and load balancing.

In this paper, we present a MapReduce based approach to accelerate SIFT feature extraction. The original SIFT feature extraction is reformed to fit the model, which can extract SIFT feature simultaneously on Hadoop[7] cluster and shows good speed up rate.

The rest of this paper is organized as follows. Section II surveys the related works. Section III briefly describes the SIFT feature extraction progress. Section IV details the MapReduce based SIFT feature extraction algorithm. Section V implements an experiment and analyzes the results. Finally, we conclude our work in section VI

II. RELATED WORK

Using parallel approaches to accelerate SIFT feature extraction have been studied extensively in recent years. In this section, we briefly discuss the work related to parallel SIFT feature extraction.

The parallel approach to accelerate SIFT feature extraction is mainly divided into two categories. One is to make use of special hardware to accelerate the feature extraction process. Bonato[8] proposes an FPGA based parallel hardware architecture to accelerate SIFT feature extraction and apply it to mobile robot localization. Kim[9] takes advantage of CUBA and SSE separately based on OpenMP to achieve parallel SIFT feature extraction. Warn[10] applies the GPU and OpenMP based SIFT parallel SIFT feature extraction to massive satellite image processing.

Special hardware acceleration approach is dependent on additional hardware, thereby increasing the cost and bringing about poor portability. In addition, to maximize the performance of the hardware, programming must acquainted with the new hardware feature. Furthermore, this type of architecture exhibits poor scalability on large number of hardware.

The other approach to accelerate SIFT feature extraction is to make use of software-based methods on general-purpose parallel computer. Hao[11] achieves a parallel extraction of SIFT based on OpenMP in the shared storage platform, which gets a 9.7~11 times and 52 times on a 16 core SMP system and a 64 core CMP system separately.

978-1-4799-2830-9/14 $31.00 © 2014 IEEE
DOI 10.1109/CLOUDCOM-ASIA.2013.22

Although software based approach can achieve high speed up, which is very expensive and number of core is unable to scale.

III. A BREIF DESCRIPTION OF SIFT

The SIFT feature extraction progress can be divided into the following steps.

A. Build Scale-Space Pyramid

To achieve scale invariance, SIFT describes the same feature at different spatial scales. The SIFT feature extraction spaces are implemented as a Gaussian scale pyramid, which is computed via convolving the Gaussian filter function with a variable core on the image. The SIFT scale-space pyramid consists of o successive octaves, with s scales per octave. The first level image of upper octave is a sub-sample of the top level image in the lower octave.

DoG(Difference of Gaussian) image pyramid is obtained by subtracting adjacent image in the Gaussian scale pyramid. The Difference-of-Gaussian (DoG) provides a good approximation for the Laplacian-of-Gaussian. It can be efficiently computed by subtracting adjacent scale levels of a Gaussian pyramid.

B. Detect SIFT Keypoint

Keypoint is detected in DoG pyramid by comparing each point with the 8 immediate points in the same space and the 9 immediate points in the upper and downer spaces. Only the extreme point is selected as keypoint. The unstable feature points are removed, which include the points susceptible to noise interference and at edge positions. Next, the position, scale and principal curvature of the keypoint are recorded.

C. Compute Feature Orientation

To achieve rotational invariance, the keypoint orientation parameter is characterized by the gradient direction distribution of the neighboring pixels. The point' gradient magnitude *m* and gradient orientation *θ* are as shown in equation.

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

$$\theta(x, y) = \tan^{-1}(L(x, y+1) - L(x, y-1)) / (L(x+1, y) - L(x-1, y))$$

Next, the histogram of oriented gradients (HOG) methods is used to determine the main orientation of feature point precisely.

D. Compute Feature Vector

First of all, the image gradient magnitude and orientation is sampled around the keypoint location. Sampling is performed in a regular grid of 16×16 locations covering the interest region. For each sampled location, the gradient orientation is involved in a coarser 4×4 grid of gradient orientation histograms with 8 orientation bins each. Once all orientation histogram items have been computed, those items are concatenated to form a single 4×4×8=128 dimensional feature vector.

IV. MAPREDUCE BASED SIFT FEATURE EXTRACTION ALGORITHM

SIFT feature is a local feature of image. In all stages of the SIFT feature extraction process, which only involves the computation of the local data, showing the good parallelism in data space. Therefore, it can take advantage of MapReduce to achieve parallel feature extraction for the large scale image. The MapReduce based parallel feature extraction process can be simply described as follows: First, the large image is split into multiple small splits. Then, the splits are distributed to the datanodes of the Hadoop MapReduce cluster. Next, each datanode executes a feature extraction task independently in map progress. Finally, the intermediate results are aggregated to obtain the final result in reduce progress. And, since SIFT feature has good locality, it does not require the global communications between SIFT feature extraction process.

A. The general MapReduce model

The general MapReduce model includes a map phase and a reduce phase. The main idea of MapReduce is to divide the original big data into many small splits and then process them on distributed machines simultaneously.

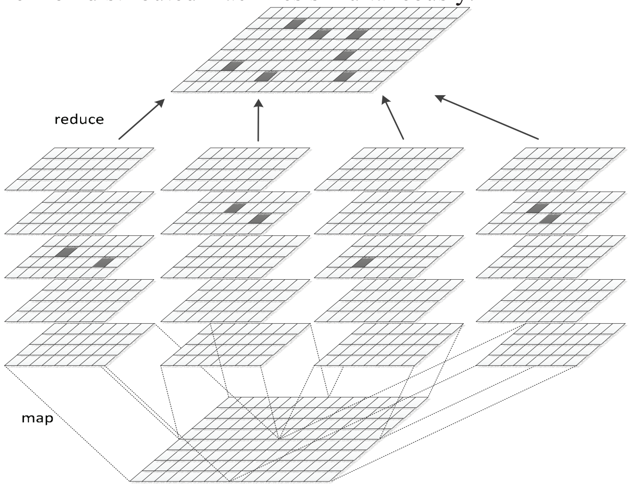

Figure 1. Figure 1 The MapReduce based feature extraction progress

As far as SIFT feature extraction is concerned, the entire image is split into multiple small splits. As shown in Figure 1, each split serves as an input split for a map task. Because the Map function takes a series of key/value pairs, processes each, and generates zero or more output key/value pairs, for SIFT feature extraction problem, the value contains partial image data, and the key is designed to "filename-x-y" styled string, the filename denotes the name of the original image, x and y denote the starting position of the original image. In each map task, the SIFT feature of a sub region in the original image is extracted. The output of map tasks is grouped by filename and the SIFT feature of the splits merged together in reduce task. The MapReduce-based SIFT feature extraction process is detailed in the following.

B. *Image Read and Write Capabilities in Hadoop*

Generally, Hadoop processes a very large data file, which contains a long sequence of atomic input records that can be processed independently [12]. The file is split automatically, normally along HDFS block boundaries. However, image data is unstructured and does not lend itself to arbitrary splitting because of the complex interrelationships between spatially adjacent pixels and correlations across multiple scales. Current-generation Hadoop doesn't provide any implicit support for image input and output formats, so a new custom input image format had to be created in Hadoop. We implemented this solution by writing new classes that extends the functionality of Hadoop API to handle image read and write capabilities. First, we extend the Hadoop's Writable interface to represent an image and its meta information. In order to read image from Hadoop, FileInputFormat and ImageRecordReader are extended to read the image file, split and extract the meta data, convert file information into (key, value) form and sends to Mapper. In order to write image to Hadoop, *ImageOutputFormat* and *ImageRecordReader* are extended to write the image to Hadoop.

C. *Image Preprocessing*

There are many types of image in real world. As for the prevalence of color images, because SIFT feature extraction is performed on the gray image, the color property makes no sense for SIFT feature extraction. So after the image is imported into Hadoop, it needs to be pre-progressed before feature extraction. The main pretreatment step is to convert the image to grayscale mode. For color image, the size is reduced to 1/3 of the original image.

D. *Image Split*

After image pretreatment, the next step is image split. Image split is a critical step in MapReduce based feature extraction. For it must ensure that the result calculated by map and reduce tasks on image splits is consistent with the result obtained from the original image. There are two key factors that affect the final result in MapReduce based feature extraction. One is to ensure that the image splits remain the parity of the original image in Gaussian scale pyramid. The other is to ensure the edge points which bring about by image split have enough boundaries in SIFT keypoint detection and feature vector computation stages.

The first level image of upper octave is a 2x sub-sampling of the top level image in the lower octave in Gaussian pyramid. The sampling process is to remove all even rows and even columns pixels in the original image. The Sampling image size is one-fourth of the original image. So when the image is split into splits, which need to remain the parity of the original image. Otherwise, the feature extracted from the image splits will be inconsistency with which extracted from the original image. For instance, as shown in Figure 2, after sampling on a 4 x 10 original image, it gets a 2x5 image. But if it is divided into two 4x5 sub images, and sampling on the sub images it will get two 2x 3 images. The result of the sampling is inconsistent with the original image, simply because the original rows and columns parity is not keeping in the divided images. Therefore, while splitting image into blocks, the blocks' height *h* and width *w* should satisfy the following equation: If SIFT scale space has *o* octaves, the blocks' height *h* should satisfy $h \bmod 2^{o-1} = 0$ and width *w* should satisfy $w \bmod 2^{o-1} = 0$. If the 0th octave' 0th level image is a 2X digital zoom of the original image, the blocks' height *h* should satisfy $h \bmod 2^{o-2} = 0$ and width *w* should satisfy $w \bmod 2^{o-2} = 0$.

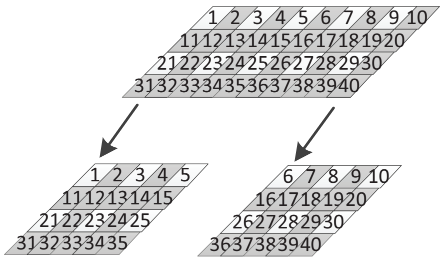

Figure 2. Figure 2 The MapReduce based feature extraction progress

In the SIFT keypoint detection stage, each point compares with 8 immediate points in the same space and 9 immediate points in the upper and downer space. When splitting the image into splits, each split cut more than 1 line as boundary to ensure that the computation result on the pixels in the new generated edge is same as the original image. So each split cut more than 8 lines as boundary to ensure that the computation result on the pixels in the new generated edge is same as the original image. So considering the above two situations, each split needs to split more than 8 lines at edge as boundary. And the feature point which computed in the auxiliary area will be removed from the final result.

E. *Feature extraction on each block*

Hadoop distribute the input split (including the image splits and program) to the datanodes of the cluster. For each input split a map task is spawned. The map tasks execute the feature extraction progress as described in independently. Finally, each map task outputs a (Key, Value) pair.

F. *Merge Feature together*

The reduce task get the information generated in Map tasks. The sift feature extracted in map tasks is grouped by filename in reduce task. The reduce task merges the SIFT feature together and maps to the original image.

V. EXPERIMENTAL RESULTS

In this section, we present the implementations of the MapReduce based SIFT feature extraction algorithm which is described in Section IV and evaluate it. The evaluation was performed on a Hadoop cluster, which includes a namenode and 8 datanodes. The namenode is a DELL Optiplex 960 PC(Intel Intel Q9650 3.0 GHz Core 2 Quad CPU, 4G DDR2 memory) with CentOS 6.4. The secondarynamenode and the namenode use the same computer. The datanodes are eight Lenovo ThinkCenter

M8380T PCs (Intel i3 540@3.07GQuad CPU, 4G DDR2 memory) with CentOS 6.4. And the Hadoop version is 1.04.

The experimental procedures take the NASA's experimental EO-1 satellite images as the experimental data, as shown in table I. The experimental procedure on Hadoop is as follows: First, we import the remote sense images in the Hadoop cluster, then start a MapReduce job and modify the configuration file to specify the number of image splits. In this experiment, the MapReduce job converts the input image into 8 bit grayscale image and split each image into 16 splits. The Hadoop cluster starts 16 map tasks to compute the feature at the same time. Finally one reduce task merges the intermediate results together and maps to the original image.

TABLE I. EXPERIMENT DATA

Image name	size	depth (bit)
EOlA137032201031311OT0_B01_LlT.TIF	6021 * 11541	16
EO1A1270212013111110TX_B01_L1GST.TIF	5941*11221	16
EO1A1410292013111110KF_B01_L1GST.TIF	6061*10621	16

In order to compute the MapReduce approach' speed up rate, we run the MapReduce based SIFT feature extraction algorithm on an eight datanodes Hadoop cluster and a stand-alone PC respectively.

The image EOlA137032201031311OT0_B01_LlT.TIF and the single machine SIFT feature extraction result are shown in Figure 3.

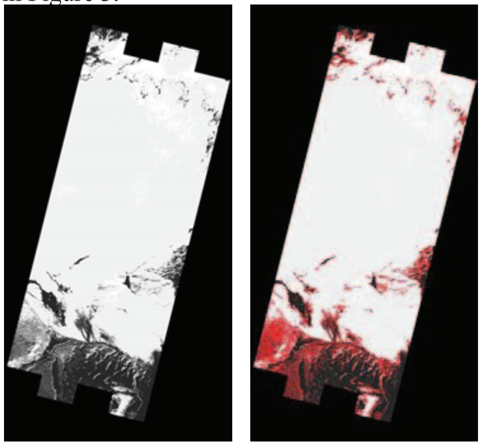
Figure 3. Figure 3 The single machine SIFT feature extraction result

The intermediate results of MapReduce based approach on EOlA137032201031311OT0_B01_LlT.TIF are shown in Figure 4.

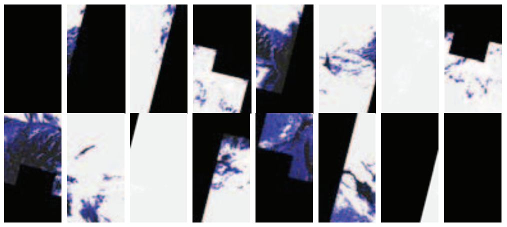
Figure 4. Figure 4 The intermediate results of MapReduce based approach

The final MapReduce based result merged by intermediate results is shown in Figure 5.

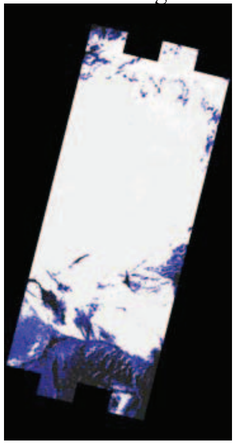
Figure 5. Figure 5 The final MapReduce based result

The feature points obtained from MapReduce based algorithm is exactly the same as which obtained from single machine, as shown in Figure 6.

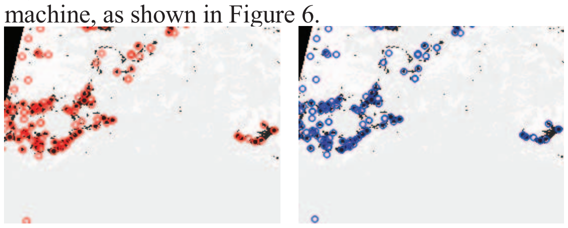
Figure 6. Figure 6 The details of two approaches

The times each approach takes are show in table II. The MapReduce based approach gets a 6.3 times on a 8 datanodes Hadoop cluster. Thus it can be seen that the MapReduce based SIFT feature extraction algorithm can greatly improve efficiency.

TABLE II. EXPERIMENT DATA

Image name	Hadoop(ms)	Single Machine(ms)
EOlA137032201031311O T0_B01_LlT.TIF	17793	112091
EO1A1270212013111110 TX_B01_L1GST.TIF	17004	109851
EO1A1410292013111110 KF_B01_L1GST.TIF	18129	121663

VI. CONCLUSIONS

We have presented a novel MapReduce based method for the SIFT feature extraction, which can extract SIFT feature simultaneously on Hadoop cluster and shows good speed up rate. We have demonstrated the practicality of our system by extracting SIFT feature on several Landsat images, confirming that the MapReduce based approach can accelerate the feature extraction progress effectively.

REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Comput. Vision, 60(2):91–110, 2004.

[2] H. Bay, T. Tuytelaars, and L. J. Van Gool, "SURF: Speeded Up Robust Features," In ECCV, pages 404–417, 2006.

[3] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors", IJCV, 1(60):63–86, 2004.

[4] J. Matas, O. Chum, M. Urban, and T. Pajdla. "Robust wide baseline stereo from maximally stable extremal regions". In BMVC, pages 384–393, 2002.

[5] K. Mikolajczyk and C. Schmi, "A performance evaluation of local descriptors," In CVPR, pages II: 257–263, 2003.

[6] J. Dean, S. Ghemawat, "MapReduce: simplified data processing on large clusters," In Proc. of the 6th Symposium on Operating Systems Design and Implementation, San Francisco CA, Dec. 2004.

[7] Apache Hadoop. http://hadoop.apache.org/

[8] V. Bonato, E. Marques, and G. A. Constantinides, "A parallel hardware architecture for scale and rotation invariant feature detection," IEEE Transactions on Circuits and Systems for Video Technology. Vol. 18, NO. 12, Dec. 2008.

[9] Junchul Kim, Eunsoo Park and Xuenan Cui, "A fast feature extraction in object Rrcognition using parallel processing on CPU and GPU," Proc. the 2009 IEEE International Conference on Systems, pp.3842-3847, 2009.

[10] Warn S, Emeneker W and Cothren J, "Accelerating SIFT on Parallel Architectures," Proc. 2009 IEEE International conference on cluster computing and Workshops, pp.577-580, 2009.

[11] Feng Hao, Eric Li and Chen Yurong, "Parallelization and characterization of SIFT on multi-core systems", Proc of IEEE International Symposium on Workload Characterization, pp.14-23, 2008.

[12] T. White, Hadoop: The Definitive Guide, 3rd ed., O'Reilly Media, May 2012.