A Hardware Implementation of Bag of Words and Simhash for Image Recognition

Shengye Wang, Chen Liang, Xuegong Zhou*, Wei Cao, Chenlu Wu, Xitian Fan and Lingli Wang
The State Key Laboratory of ASIC and System, Fudan University
Shanghai, China 201203
*Email: zhouxg@fudan.edu.cn

***Abstract*—Algorithms such as Bag of Words and Simhash have been widely used in image recognition. To achieve better performance as well as energy-efficiency, a hardware implementation of these two algorithms is proposed in this paper. To the best of our knowledge, it is the first time that these algorithms have been implemented on hardware for image recognition purpose. The proposed implementation is able to generate a fingerprint of an image and find the closest match in the database accurately. It is implemented on Xilinx's Virtex-6 SX475T FPGA. Tradeoffs between high performance and low hardware overhead are obtained through proper parallelization. The experimental result shows that the proposed implementation can process 1,018 images per second, approximately 17.8x faster than software on Intel's 12-thread Xeon X5650 processor. On the other hand, the power consumption is 0.35x compared to software-based implementation. Thus, the overall advantage in energy-efficiency is as much as 46x. The proposed architecture is scalable, and is able to meet various requirements of image recognition.**

***Keywords*—Bag of Words; Simhash; image recognition; FPGA implementation**

I. Introduction

Image recognition has been a hot topic due to researches in computer vision and booming of the Internet. As the number of images on the Internet increase rapidly, there is great demand for efficient image searching scheme in huge databases. There have been some algorithms developed to satisfy such harsh demands. For example, *SURF*, standing for *Speeded Up Robust Features* [1], is one of them, which creates descriptors for images based on local features. Methods like *Bag of Words* [2] are exploited to simplify the process of comparing two descriptors. To reduce size of the database and calculation during comparison, *Simhash* [3] have been developed based on Bag of Words representation. The similarity of two images is preserved as the similarity between two fingerprints. Most implementations of these algorithms are based on software. In some systems, such implementations could be a bottleneck under given requirements, such as performance demand or power constraint. In this paper, a hardware architecture design is proposed to accelerate these algorithms.

The proposed architecture of Bag of Words and Simhash focuses on generating the fingerprint of an arbitrary image on line. The system accepts image descriptors as its input, which is a set of local feature descriptors created by SURF algorithm. With pre-calculated dictionary, the set of features is compressed into a k-bit fingerprint using Simhash algorithm, where k is an integer and usually power of 2. Bag of Words algorithm is exploited in this procedure, mapping an arbitrary interest point to the most similar entry in the dictionary. According to the property of Simhash, the Hamming distance between two of such fingerprints can represent the difference of the images. Then, the nearest match is searched in a huge fingerprint database. The proposed architecture is implemented on Xilinx's Virtex 6 high-performance FPGA. With our proposed architecture and considered tradeoffs, the energy-efficiency is improved by 46x compared to software-based implementations.

In section II, a brief introduction to the target algorithms is presented. In section III, a hardware architecture designs for such algorithms is proposed. Some tradeoffs have been made for easier implementation on hardware without lowering the accuracy. In section IV, the implementation details in FPGA are reported. In section V, the profile of the proposed implementation is presented. Performance and energy consumption are compared with CPU-based system. The advantage of the proposed system is presented in section VI as conclusion.

II. Background of Algorithms

Fig. 1 shows the dataflow for proposed implementation. SURF module is not included in the design, making the design compatible with other image descriptor generator. Bag of Words and Simhash, which are the core parts of the flow, are implemented in hardware. A match module is also included to find a match in the database.

A. Speeded Up Robust Features

Speeded Up Robust Features, also known as *SURF*, is a performant scale- and rotation-invariant interest point detector and descriptor [1]. There are several variations of SURF algorithm implementation, which share the same idea but differ in data representations. In the proposed architecture, image descriptors from SURF are used as input.

B. Bag of Words

To compress all the n vectors into a k-bit string, known as fingerprint of an image, *Bag of Words* algorithm proposed by

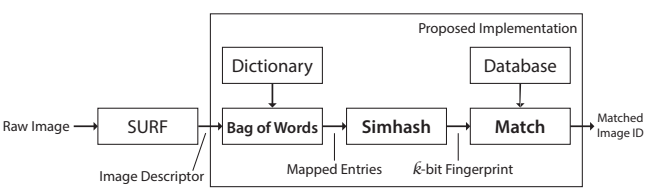

Fig. 1: Dataflow diagram for the proposed architecture.

978-1-4799-2198-0/13/$31.00 ©2013 IEEE

Sivic et al [2] is performed. Given a set of feature descriptors, choose a typical subset of m as a 'dictionary'. All features can be mapped to the closest entry in this dictionary. Common occurrence of a 'word' in the dictionary indicates the images are similar. Also the total frequency of a particular word is used as a measure of its distinctiveness. If two images share an identical rare word, they are more likely to contain the same object. The major calculation is to find out the closest match in the dictionary for each interest point. The distance is defined as Manhattan distance. That is, given two d-dimensional vectors, $\mathbf{a}$ and $\mathbf{b}$, the distance is defined in (1).

$$dis(a,b) = \sum_{i=0}^{d-1} |a[i] - b[i]| \tag{1}$$

C. Simhash

Simhash [3], a fingerprint technique used in search engines, is one of the candidates which reduce the size of an image. The basic idea is to create k-bit fingerprint for each image, with such a property: the Hamming distance between two fingerprints can represent the difference between two original images.

Each entry in the dictionary holds a unique k-bit string and a number. The number, also known as 'weight', indicates the importance of that entry, which is a measure of distinctiveness. Each interest point of the n features in an image can be mapped to an entry in the dictionary. Suppose the weight of an entry is w, the k-bit string can be expanded to a k-dimensional vector of numbers. If a bit is 0, the value of the corresponding dimension is w, otherwise it is $-w$. These n vectors are summed up. The string composed of the sign bits of each dimension in the sum, k bits in all, is the fingerprint. Based on the description above, images with similar features will generate similar fingerprints. In other words, if most of the features are similar in both images, the Hamming distance would not be large even some interest points do not match.

D. Dictionary Creation

There are several methods to create a dictionary. K-means clustering algorithm [4] is exploited in the proposed procedure. Suppose K entries are expected in a dictionary, K points in the space are randomly chosen as initial cluster centers. Each word in the space is classified to a cluster with the closest center nearby. After all the words are processed, the arithmetic mean of all the vectors in the same cluster are calculated and used as the new center. After a few iterations, the cluster centers will almost stay still, and could be used as entries in the dictionary. The dictionary can be generated offline. With one time creation, it can be used ever later.

As mentioned above, every entry in the dictionary also includes a unique k-bit string for Simhash calculation. The k-bit hash can be created with any method. In our implementation, md5 algorithm is applied. An additional number representing the weight of a word is required. According to *TF-IDF* method [5], it is calculated with $w = \log(D/j)$, where D represents the total number of images used to create the dictionary, while j is the number of images which hold an interest point in the corresponding cluster. Therefore, a less frequent entry will have a greater weight and more influence on the final fingerprint.

III. Proposed Architecture

The proposed architecture is composed of 3 parts. A fingerprint unit generates k-bit hash value for a given image. A match unit finds out the closest match in the database. An I/O unit provides the interface for data input and output.

A. k-bit Fingerprint Generation

An image descriptor contains n descriptors of local features, where n can be an arbitrary number. To improve the performance, distance between interest points and entries in dictionary are calculated in parallel. In other words, n units calculate the Manhattan distance simultaneously. Therefore, the list of features should have been trimmed to a reasonable size during generation of the image descriptors [6], so that parallelization can be possible.

1) Calculation Unit: Because there are n calculation units, a single unit should not occupy large amount of resource. Besides, the interconnection among them can not be complicated or there would be excessive routing burden. Considered these factors, tradeoffs between time and area are made. Instead of calculating the Manhattan distance between the vectors per cycle or in d cycles, it is done in d/p cycles, where d is the number of dimensions and p is the degree of parallelization. Both d and p are power of 2, $d \geq p$. Therefore, (1) is transformed to (2):

$$dis(a,b) = \sum_{i=0}^{d/p-1} \sum_{j=0}^{p-1} |a[i \times p + j] - b[i \times p + j]| \tag{2}$$

In (2), the d/p accumulations are executed in serial with an accumulator, and the p accumulations are executed in parallel with a p-input pipelined adder tree. The differences of p dimensions between a dictionary entry and an interest point are calculated respectively by subtracting and then calculating the absolute value at the same time. Additional registers are placed between them to ensure the register-to-register delay is short. Then a pipelined p-input adder tree sums them up to get sum of the p dimensions. Since there are d/p accumulations, an accumulator at the end adds all the d/p partial sums up to get the difference between both vectors. An example is shown in Fig. 2 when $d = 64$ and $p = 4$.

2) Parallel Calculation: As shown in Fig. 3, n units calculate the Manhattan distance simultaneously. The dictionary entries and the interest point descriptors are not broadcasted to every calculation unit to avoid excessive routing burden, which may lead to high hardware overhead and poor maximum frequency. Before calculation starts, dictionary and interest points are shifted to the chain, filling the calculation unit one after another. Calculation unit only calculates the distance between a dictionary entry and the local feature as defined in (2). The closest match, as well as the corresponding dictionary entry, is preserved for Simhash. Each calculation unit is accompanied with a comparator, which updates the current match.

Therefore, the number of total cycles consumed in looking up the nearest dictionary entry is $t \times d/p$, where t is the number of entries of dictionary, d is the dimension of each interest point and p is the degree of parallelization. Larger p will decrease the calculation time of generating the k-bit fingerprint at the cost of more area.

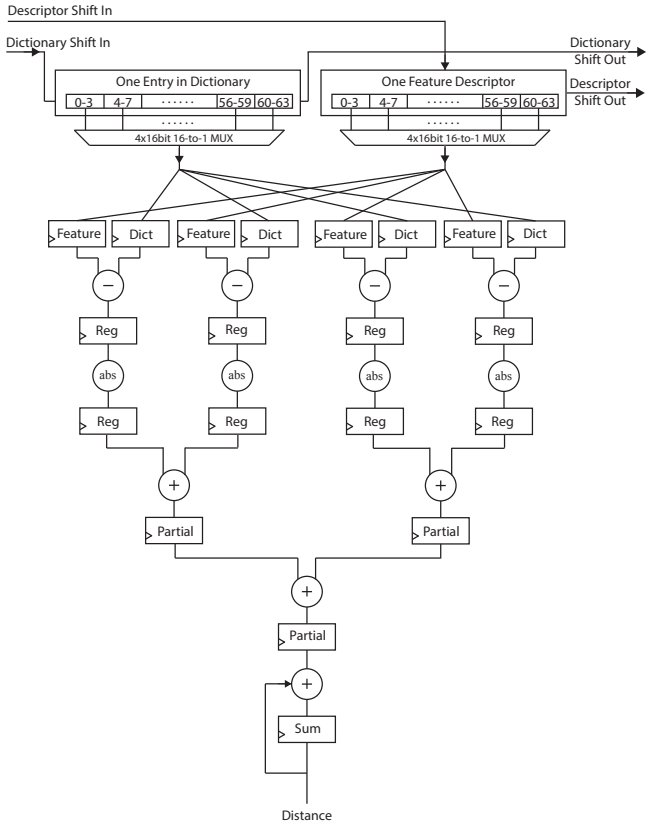

Fig. 2: Structure of calculation unit, when $d = 64$ and $p = 4$.

3) Serial Calculation: Simhash, the final procedure, is to sum all the best matches up. Since it is not the most time-consuming procedure, serial calculation can be exploited to save hardware resource. The registers hold matched dictionary entries are chained. While they shift to the right, an accumulator adds the expanded vectors up according to the procedure illustrated in section II-C. Once the shifting is done, the string of k sign bits of the sum is the fingerprint according to Simhash algorithm. Serialization will only increase the latency of processing an image by at most n clock cycles, which can be neglected in further optimization. Because the calculation units can be utilized to process successive image during shifting, throughput is not reduced.

B. Closest Match Search

In the proposed system, the closest match is searched in the whole database to ensure predictable performance. That is, the Hamming distance between the generated fingerprint and the best match in the database is minimized. A pipelined adder tree is exploited to obtain the Hamming distance between the Simhash result and a database record. To calculate the Hamming distance, 'exclusive or' is performed over two k-bit strings to get one k-bit vector representing the difference of both vectors. The intermediate result is registered before the adder tree counts the number of 1's in the vector. Because of pipelining, one database record can be processed in one cycle on average.

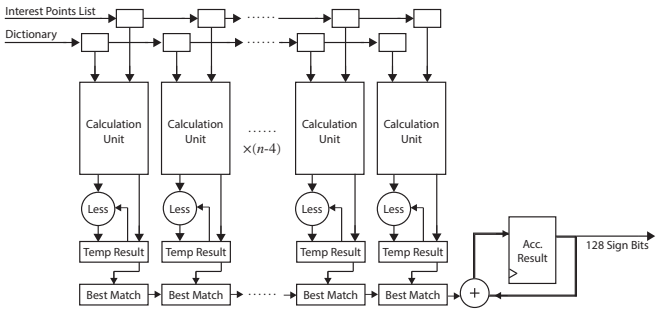

Fig. 3: Structure of k-bit simhash calculation circuit.

C. Input/Output Interface

Two individual memory interfaces are necessary for the proposed architecture. One is for dictionary storage and the other is for database storage. To make the access simple, some glue logic is introduced to the structure, which could provide sequential access to the memory. The glue logic hides the complication of operating memory controllers, exposing FIFOs (First In First Out) to the fingerprint unit and the match unit.

Data are exchanged through Ethernet to ensure flexibility. A specialized protocol over UDP (User Diagram Protocol) is built for communication. Basic operations include downloading dictionary and database, sending to-be-matched images and returning the match results. The protocol is simple and efficient for local area network because of the advantages of UDP.

IV. FPGA Implementation

A. System Scale

The number of interest points in an image is limited in our implementation, i.e. $n \leq 100$. For most of the images, 100 interest points are enough to distinguish it from another. Less important interest points can be trimmed with the method proposed in [6].

A 64-dimensional normalized vector represents a single interest point, i.e. $d = 64$. 4-way parallelization is exploited to accelerate the calculation while keeping low hardware overhead, i.e. $p = 4$. 16-bit fixed-point number is exploited to represent each dimension. Fixed-point representation is reasonable because the vector is normalized, which means that all dimensions are between -1.0 and 1.0. The precision loss can be safely ignored because every interest point will be mapped to one of the entries in the dictionary, which are few and far between.

A dictionary with 5,432 words is exploited in the implementation, i.e. $m = 5{,}432$. It is created using K-means algorithm with 3,994,534 interest point samples, which are extracted from 9,812 images. The database contains 9,812 fingerprints, which has been pre-calculated with high-performance computer. More records can be added into it on line. Because fingerprint generation and matching are parallel, the database can contain much more records than 9,812 without affecting the performance.

Xilinx's Virtex 6 FPGA XC6VSX475T is exploited in the implementation. It is equipped with 74,400 logic slices, 2,016 DSP blocks and numerous other resources. It is also

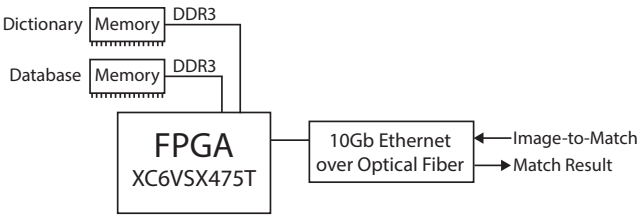

Fig. 4: FPGA and its interfaces.

the highest-end model for computing-intensive application in Virtex 6 family. The clock frequency of the Ethernet I/O unit is 156.25 MHz, which is the same frequency as the main clock to avoid cross clock domain complication.

B. System Infrastructure

The proposed implementation is integrated into a hybrid system oriented for high performance computing. To evaluate the performance and the energy-efficiency, our sub-system includes one FPGA and one server. A server distributes the computing tasks, i.e. to forward image descriptors to the FPGA. The FPGA does the calculation and matching before returning the result to the server. The program on server also profiles the performance of the FPGA. The organization of on-board hardware is shown in Fig. 4. Those DRAMs are both 8 GB in the system. However, only a few mega bytes are occupied because the application is only computationally intensive. The dictionary and the fingerprint database are small in size.

V. Experimental Results

Resource usage is shown in table I. Note these numbers also include I/O interface, such as Ethernet media access controller, DDR3 memory controller, etc. A test suite containing 9,812 images with 9,812 records in the fingerprint database is exploited for evaluation. The experiments show that the results generated by the software and hardware are identical, indicating the correctness of hardware implementation.

The CPU-based profile is obtained from a modern multi-core server, which is based on Intel's 6-core Xeon X5650 processor running at 2.66 GHz, with 32 GB memory. The 32 GB memory is able to hold the whole dictionary and the fingerprint database. The CPU provides 12-thread parallelism, which is fully exploited to utilize the power of the processor to the maximum extent: 12 threads matches interest points with words in the dictionary simultaneously. The time and power consumption of both implementations are shown in table II.

TABLE I: Resources Utilization

Resoruces	Utilization	Percentage
Slice Registers	225,054 / 595,200	37%
Slice LUT	170,813 / 297,600	57%
Slice LUT Used as Logic	77,781 / 297,600	26%
Slice LUT Used as Memory	22,970 / 122,240	18%
Slice LUT Used as Routing	70,062 / 297,600	24%
Occupied Slices	48,399 / 74,400	65%
Bonded IOBs	256 / 840	30%
BRAM RAMB36E1/FIFO36E1s	44 / 1,064	4%
BRAM RAMB18E1/FIFO18E1s	4 / 2,128	1%
BUFG/BUFGCTRLs	12 / 32	37%

TABLE II: Energy Efficiency Comparison

	FPGA-based	CPU-based	Ratio
Number of Images	9,812		-
Maximum Number of Features	100		-
Size of Feature	64×16bits		-
Size of Dictionary	5,432		-
Fingerprints in Database	9,812		-
Total Time (s)	9.632	171.672	0.0561
Frames Per Second	1018.7	57.2	17.81
Full Load Power (W)	58.96	167.99	0.3510
Frames Per Joule	17.3	0.379	45.65

As shown in table II, the proposed system improves the performance by 17.8x, which is 1,018 frames per second, compared to implementation on modern multicore processors, which can only process 57 frames per second. The FPGA board consumes 59 Watts power on average, which is 0.35x compared to CPU-based system, i.e. 168 Watts. Therefore, the overall energy-efficiency is 46x better.

VI. Conclusion

Bag of words and Simhash algorithms are widely used in computer vision and machine learning. However, most of the current implementations are based on software. In this paper, a hardware-based scheme is proposed to accelerate the calculation. With the proposed architecture, tradeoffs between high performance and low hardware overhead are obtained through the proper parallelization.

The proposed architecture design is scalable. Better performance can be achieved at the cost of more hardware resources. The proposed implementation is able to meet the requirement for applications like images searching, filtering or recognition in Internet, robotics, transportation, medicine, etc.

Acknowledgment

The authors would like to thank Qian Yu, Huotian Zhang, Yijing Sun and Zimu Li from Department of Microelectronics Fudan University for productive (and challenging) discussions. This paper is supported by National Natural Science Foundation of China (61131001, 61171011) and Fudan's Undergraduate Research Opportunities Program.

References

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[2] J. Sivic and A. Zisserman, "Efficient Visual Search of Videos Cast as Text Retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 4, pp. 591–606, Apr. 2009.

[3] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, May 2002, pp. 380–388.

[4] J. B. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," Defense Technical Information Center, 1966.

[5] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[6] P. Turcot and D. G. Lowe, "Better matching with fewer features: The selection of useful features in large database recognition problems," in *IEEE 12th International Conference on Computer Vision Workshops*, 2009, pp. 2109–2116.