GPU-Based Parallel Programming for Digital Document Image Classification

Refi Sencia Dity, James Purnama, Reggio N. Hartono, Maulahikmah Galinium
Department of Information Technology, Faculty of Engineering and Information Technology
Swiss German University, Indonesia

Keywords:

Multithreading
Document Images Analysis
Document Images Classification
CUDA C
GPU
Pthreads

ABSTRACT

The use of document images is currently increasing. It creates another need to classify document images into their type automatically. To recognize the document images and classify the type, several feature extraction methods have been used in this research. Binary Morphological erosion with 9 intersection types is one of the simple and effective feature extraction method. Due to time processing, this method is not applicable. Therefore, the purpose of this research is focusing on the use of multithreading in multi-core processor with POSIX Threads (Phtreads) and multithreading GPU-based with NVIDIA CUDA in order to speed up the processing time. The results show that with local processing, the classification accuracy can be improved. Furthermore, Pthreads implemation in dual core processor can increase the processing speed around 2 scale factor. In the other hand, CUDA implementation can increase the processing speed around 100 scale factor compare with single thread program and 60 scale factor compare with multithreading in dual-core processor.

Corresponding Author:

Maulahikmah Galinium,
Departement of Information Technology, Faculty of Engineering and Information Technology,
Swiss German University,
EduTown BSD City, Tangerang, Indonesia.
Email: maulahikmah.galinium@sgu.ac.id

1. INTRODUCTION

The increasing use of digital document images currently becomes the first concern of this research. The growing amounts of internet users, the widespread use of office automation system and space problem are some of the reasons why people tend to move to digital document images. Furthermore, there are already a lot of research in the area of processing and maintaining digital document images. The fact that there are already many industries which migrating their paper documents into digital documents creates another important need to classify document images according to their type. This classification system will increase the efficiency in digitalizing and archiving digital document images. In general there are 2 approaches to analyze digital document images. First approach is by its text contents (Textual Processing). In textual processing, digital document images analysis will be based on its text components. The second approach is called graphic processing. It deals with the non-textual line and symbol components that make up line diagrams, delimiting straight lines between text sections, company logos, etc.[1]. The classification method that is proposed in this research will be based on graphic processing.

When developing a methodology for document classification, beside accuracy, speed also needs to be considered since some existing classification methods are not applicable because of the speed problem even though they can yield high classification accuracy. How to increase processing speed using multithreading parallel programming approach becomes the focus of this research. There are 2 approaches that will be discussed in this research. The first approach is the use of multithreading in multi-core processor while the second approach is the utilization of graphics processing unit (GPU) for general-purpose, or popularly called General Purpose Computing on GPU (GPGPU). It means that we utilize GPU function not only for rendering tasks but also for executing more general computing tasks. A few years ago, GPGPU was said to be a complicated task because the programmer needs to learn new GPU architectures, but with the

introduction of a high-level API from NVIDIA called Compute Unified Device Architecture (CUDA), we can exploit the power of GPGPU using simple C code.

The purpose of this research is to develop a simple document image classification method based on binary morphological erosion with 9 intersection types with local processing and the focus of this research will be the use of multithreading in multi-core processor with POSIX Threads (Pthreads) and GPU based parallel programming with NVIDIA CUDA technology to speed up processing time.

2. RESEARCH METHOD

Design and implementation of this research follows the following steps: sample of acquisition, binarization, feature extraction and classification. Document samples gathered from IT department office in SGU. We choose 3 different types of document, from each type, we took 10 samples. Those 3 types of documents are attendance list document (type 0), unit of subject delivered document (type 1), and staff request to Information System Service (ISS) department document (type 2). To process the document, each document needs to be converted into digital image by using scanner machine. Furthermore, binarization is conducted. The first step that should be done in binarization is to determine the threshold value. This following iterative algorithm for global thresholding [2] is used in order to find the best threshold:

1. Select an initial estimate for global threshold, T. In this research, average image intensity is chosen as initial estimate for T.
2. Segment the image using T. This will produce two groups of pixels: G_1 consisting of all pixels with intensity values > T, and G_2 consisting pixels with values ≤ T.
3. Compute the average (mean) intensity values m_1 and m_2 for the pixels in G_1 and G_2, respectively.
4. Compute a new threshold value: T = (m1 + m2)
5. Repeat Steps 2 through 4 until the difference between values of T in successive iterations is smaller than a predefined parameter ΔT.

Furthermore, feature extraction steps are also conducted as follow:

1. We use 9 intersection types to recognize the line structure of document image, as shown in figure 1.

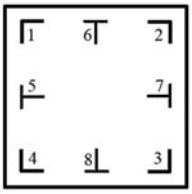

Figure 1. Representation of 9 intersection types

2. For the intersection locations, binary morphological erosion with 9 structuring elements, one for each intersection model is used. For each intersection types, we create intersection model of size 71 pixels (35 pixels on each leg) [3][4]. To reduce time and save memory, only 4 intersection types are used (Intersection types 1,2,3 and 4), this approach is possible because the remaining intersection (Intersection types 5,6,7,8 and 9) can be get from the combinations of the first four types [5]. Called IMGi (i = 1 to 9) as the result of the erosion by the structuring element i, then :

$$IMG5 = IMG1 \cap IMG4 \qquad IMG6 = IMG1 \cap IMG2$$
$$IMG7 = IMG3 \cap IMG2 \qquad IMG8 = IMG3 \cap IMG4$$
$$IMG9 = IMG1 \cap IMG3 \quad \text{or} \qquad IMG9 = IMG2 \cap IMG4$$

Because we only keep track of the amount of the 9 intersection types that are contained in the document, there might be a case where 2 document images have the same amount of intersections. However, these intersections in each document are located in different part of the documents; this condition can cause ambiguity when it comes to classification. Therefore, to reduce ambiguity, we divide document images into several regions, then extracting the features in each region and compare the result locally. Figure 2 shows example of image division, in this example document image divide into 2 vertical regions and 2 horizontal regions, with a total of 4 regions. Implementing the method in standard C code turned out to be ineffective because of the time it took to process the images, especially in binary morphological erosion operation. In an effort to increase processing speed, implementation of multithreading for binary morphological erosion operation is proposed, for this research, 2 different approaches of multithreading are conducted: using Pthreads and CUDA C. In the implementation of multithreading for extracting the document images features,

the problem of race condition does not appear because each pixel is processed independently and are not related with each others.

Implementing Pthreads in a UNIX includes a pthread.h header/include file and a thread library. For compiling multithreaded program, we can use GCC compiler. In implementing Pthreads, programmer needs to ensure that each threads has already finished their work before the main thread exit. Pthread_join function is used to accomplish this condition. This function will wait until all threads re-join the main thread. These following steps are used to conduct feature extraction using Pthreads:

1. Create intersection type 1 and use it as structuring element
2. Divide document image vertically by the *n* threads, resulting *n* regions
3. Applying binary morphological erosion in each region concurrently with intersection type 1 as structuring element, each thread handle 1 region.
4. Combine the result from each region
5. Repeat step 1-4 for intersection type 2,3, and 4

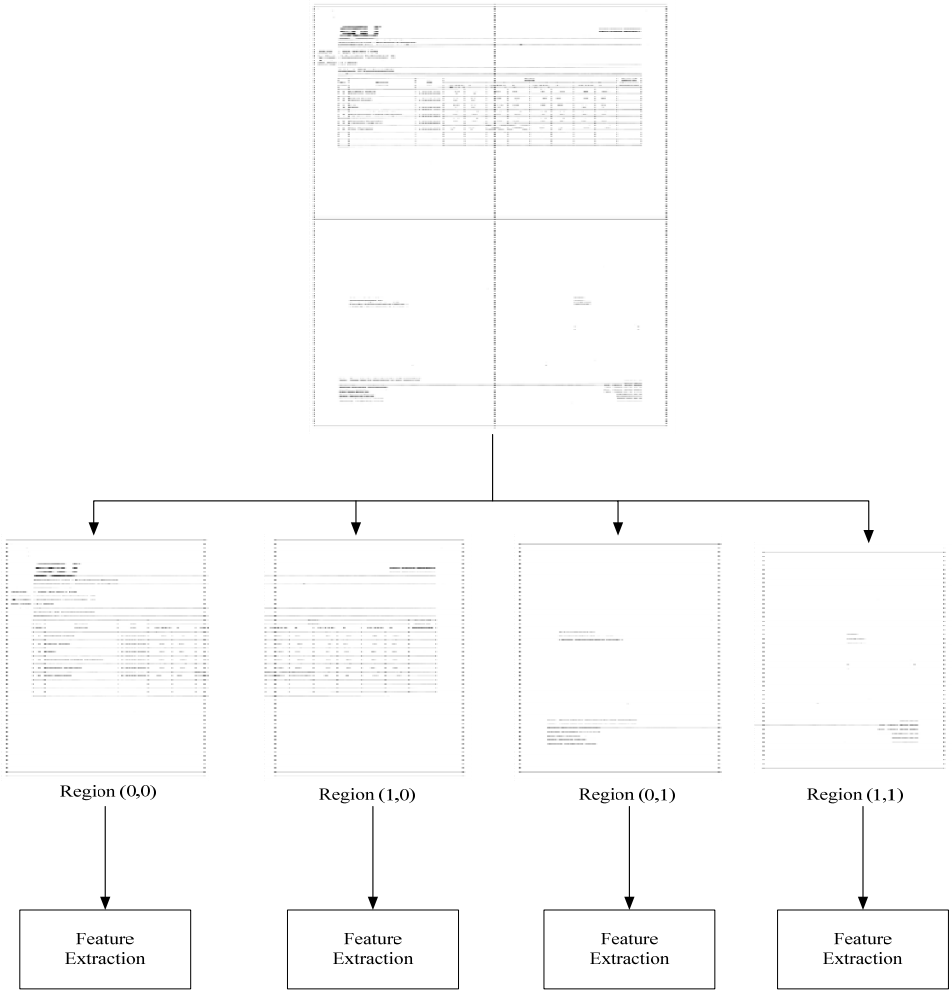

Figure 2. Local Processing

To compile a program that implements CUDA C, NVCC compiler is used. This compiler is included in NVIDIA CUDA toolkit and *.cu* file extension must be used in order to compile the program. In general, the CPU and the system's memory are referred as the host while the GPU and its memory are referred as the device. The most important step in writing CUDA C is to determine the dimension of grid and block. The arrangement of the dimension needs to be considered properly, because it can affect the effectiveness of parallel programming. These following steps are used to implement feature extraction:

1. Create intersection type 1 and use it as structuring element
2. Allocate memory and transfer the respective data to device memory

3. Applying binary morphological erosion on device using intersection type 1, with the amount of threads equal to the amount of pixels in respective region, each thread handle one pixel.
4. Get the result from device to host
5. Repeat step 1-4 for intersection type 2,3, and 4

The classification process will be based on the amount of 9 intersection types that can be found in the document. Classification process will be done using Euclidean distance nearest neighbor. The dataset for classification training is created using a flat file database. For creating database, a document template from each of document type is prepared. Then the next step is to apply feature extraction of each region using the method proposed and the result will be written into a text file as the database. This following steps are used for classify document into its type.

1. Calculate the Euclidian distance between Database and feature extraction result in each region
2. Get the average Euclidean distance from all regions.
3. Find the smallest average Euclidean distance from database

3. RESULTS AND ANALYSIS

Some experiments were conducted in order to find out the classification accuracy and also to compare between 2 different approaches of multithreading, which are multithreading based on multi-core processor using PThreads and GPU based multithreading using CUDA C.

3.1. Classification Accuracy Test

In this test, each sample document will be processed and classified into its type. Accuracy test will be based on how many document can be classified into its right type. Table 1 represents the result of classification accuracy test without local processing. The highest accuracy is reach by document type 0.

Table 1. Accuracy test without local processing

Doc Name	Doc Type	Classification Accuracy
Attendance List	0	80%
Unit of Subject Delivered	1	0%
Information System Service Form	2	60%

Table 2. Classification Accuracy test with local processing

Vertical Region	Horizontal Region	Classification Accuracy		
		Attendance List	Unit of Subject Delivered	ISS Form
1	1	80%	0%	60%
	2	80%	0%	90%
	3	30%	0%	90%
	4	80%	10%	90%
	5	50%	70%	90%
2	1	70%	10%	70%
	2	70%	40%	90%
	3	30%	70%	90%
	4	70%	90%	90%
	5	40%	90%	90%
3	1	70%	10%	90%
	2	70%	40%	90%
	3	30%	80%	90%
	4	60%	90%	90%
	5	40%	90%	90%
4	1	70%	80%	90%
	2	70%	80%	90%
	3	30%	90%	90%
	4	70%	100%	90%
	5	40%	90%	90%
5	1	70%	20%	100%
	2	70%	40%	100%
	3	70%	40%	100%
	4	70%	40%	100%
	5	50%	40%	100%

Table 2 shows the result of classification accuracy with local processing. From 25 experiments with different image region division, a total 16 region with the division of 4 vertical regions and 4 horizontal regions gain the best result. With 70% classification accuracy for document type 0, 100% for document type

1 and 90% for document type 2. However, the result of this classification accuracy test in this research need to considered more because lack of testing set, but through this research, we can concluded that with local processing, a higher accuracy can be gained.

Local processing is used in order to gain a little information about the location of the intersections and how they are arranged, therefore, it reduce ambiguity. In dividing document images into several regions to be process locally, we need to do several tests to find which image division can gain better classification accuracy. Another consideration is use better classification method to get higher classification accuracy. Some classification methods that can be considered are: Naïve Bayes, Support Vector Machine (SVM), Decision Tree, and Artificial Neural Network (ANN).

3.2. Performance Test

The aim of performance test is to measure the processing time that is needed to finish the execution of the program. This test is important in order to analyze the performance between the two multithreading approaches. Figure 3 shows the performance test of multithreading in dual core processor using Pthreads. There is a significant decrease of processing time for 2 concurrent threads and a slight decrease of time when the number of concurrent threads increases to 4. However, after the number of concurrent threads reach 4, the increasing number of threads (6,8,10,12,14,16,18,20) does not give significant impact and stay stable in about 300 seconds.

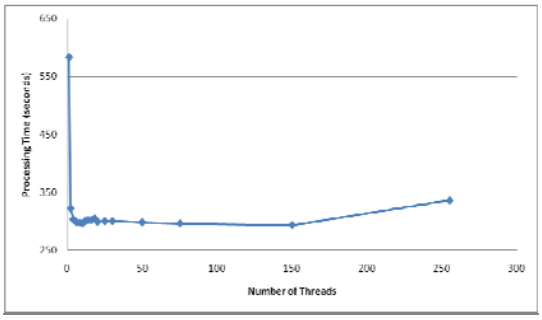

Figure 3. The graph of Pthreads performance test

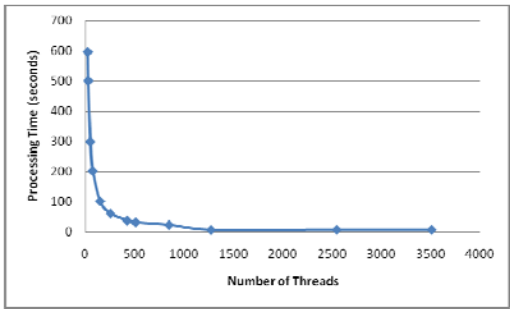

Figure 4. The graph of CUDA performance test

Figure 4 shows the CUDA performance result using *NVIDIA*® GeForce® *GTS 450* that have 192 Streaming processors and 4 streaming multiprocessors. The processing speed continues to decrease along with the increasing number of threads until it reach around 5 seconds of processing time and start to become stable. The result shows that CUDA can increase the processing speed around 100 scale factor compare with single thread program and 60 scale factor compare with multithreading in dual-core processor. The fundamental architecture design of multi core processor and many-core GPUs is the reason why there is such a large performance gap. CPU cores are designed to execute a single thread of sequential instructions with maximum speed, however GPUs are designed for fast execution of many parallel instruction threads. When we comparing the processing speed between dual core processor and many-core GPUs, in case of 30 concurrent threads, dual-core processor give a better processing speed compare with many-core GPUs. From this result, it can be concluded that the key to high performance of many-core GPUs is not in the processor clock but it lies on the use of a huge number of streaming processors. When the amount of threads is increasing while dual-core processor cannot give a performance increase, GPU can still show a significant increase of speed. In conclusion, GPU is a good alternative for implementing multithreading especially in working with image processing.

4. CONCLUSION

There are several conclusions can be made regarding this research. The methodology of using binary morphological erosion with 9 intersection types is said to be time consumed and might be not applicable in the real system. Furthermore the results of classification accuracy show that the feature extraction method using binary morphological erosion with 9 intersection types is very sensitive to artefacts. Therefore, to gain a better result, artefacts removal should be done.

Local processing is one of the simplest and effective ways to increase classification accuracy because with local processing, we can have a little information about the location of the intersections and how they are arranged in a document images, therefore, it reduces ambiguity. In dealing with multithreading with dual core processors, an huge number of threads will not really affect the processing time because the system only have 2 core processor to process every threads. With CUDA C, we can exploit the GPU to increase performance speed when dealing with such a huge data processing.

ACKNOWLEDGEMENTS

We would like to thanks to the Swiss German University to support this research with all facilities used in the research.

REFERENCES

[1] L. O'Gorman and R. Kasturi, "Document Image Analysis", IEEE Computer Society Executive Briefings, 2009.
[2] C.R. Gonzalez and E.R. Woods, "Digital Image Processing", Pearson Education, Inc, 2010.
[3] L.A.P. Neves, J.M. Carvalho, and J. Facon, "Bit Block Transfer and Structuring Element Decomposition for Table-form Physical Structure", Proceedings of SIBGRAPI 2003 – XVI Brazilian Symposium on Computer Graphics and Image Processing, São Carlos, 2003.
[4] L.A.P. Neves, J.M. Carvalho, and J. Facon, "Recognition of Deteriorated Table-form Documents: A New Approach", Proceedings of SIBGRAPI 2003 - XVI Brazilian Symposium on Computer Graphics and Image Processing, São Carlos, 2003.
[5] S.L. Taylor, F. Richard and P. Jon, "Extraction of Data from Preprinted Forms", Journal of Machine Vision and Applications, vol.5, pp.211-222, 1992.

BIBLIOGRAPHY OF AUTHORS

	Refi Sencia Dity, S.Kom, B.Eng Education: Bachelor Degree: Swiss German University, Tangerang, Indonesia Occupation: IT Professional
	James Purnama, S.Kom, M.Kom Education: Bachelor Degree: Budi Luhur University, Tangerang, Indonesia Master Degree: Swiss German University, Tangerang, Indonesia Occupation: Lecturer of Information Technology Department and Head of SPQA Department at SGU
	Reggio Nurtanio Hartono, S.Kom, M.Kom Education: Bachelor Degree: Swiss German University, Tangerang, Indonesia Master Degree: University of Indonesia, Jakarta, Indonesia Doctoral Degree: University of Auckland, Auckland, New Zealand (in process) Occupation: Lecturer of Information Technology Department at SGU.
	Dr. Maulahikmah Galinium, S.Kom, M.Sc Education: Bachelor Degree: Swiss German University, Tangerang, Indonesia Master Degree: Lunds Universitet, Lund, Sweden Doctoral Degree: University of Rome Tor Vergata, Rome, Italy Occupation: Lecturer and Deputy Head of Information Technology Department at SGU