Malaysian Journal of Mathematical Sciences 16(2): 363–377 (2022)
https://doi.org/10.47836/mjms.16.2.12

Malaysian Journal of Mathematical Sciences

Journal homepage: https://einspem.upm.edu.my/journal

Performance of CPU-GPU Parallel Architecture on Segmentation and Geometrical Features Extraction of Malaysian Herb Leaves

Hadi, N. A.*[1], Halim, S. A.[1], Lazim, N. S. M.[1], and Alias, N.[2]

[1]*Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Malaysia*
[2]*Faculty of Science, Universiti Teknologi Malaysia, Malaysia*

E-mail: normi@fskm.uitm.edu.my
**Corresponding author*

Received: 15 July 2021
Accepted: 22 April 2022

Abstract

Image recognition includes the segmentation of image boundary, geometrical features extraction, and classification is used in the particular image database development. The ultimate challenge in this task is it is computationally expensive. This paper highlighted a CPU-GPU architecture for image segmentation and features extraction processes of 125 images of Malaysian Herb Leaves. Two (2) GPUs and three (3) kernels are utilized in the CPU-GPU platform using MATLAB software. Each of herb image has pixel dimensions 1616×1080. The segmentation process uses the Sobel operator, which is then used to extract the boundary points. Finally, seven (7) geometrical features are extracted for each image. Both processes are first executed on the CPU alone before bringing it onto a CPU-GPU platform to accelerate the computational performance. The results show that the developed CPU-GPU platform has accelerated the computation process by a factor of 4.13. However, the efficiency shows a decline, which suggests that the processors utilization must be improved in the future to balance the load distribution.

Keywords: CPU-GPU; Parallel computing; Malaysian herb leaves; features extraction; image segmentation.

1 Introduction

Nowadays, most people choose herbal plant medicines or products to improve their health conditions. Some people are still practising using herbal plants to treat some diseases that modern medication cannot cure, or they prefer to use them to maintain their health. The plants are notorious for healing various types of conditions. Identification of plant species gives advantages, especially for medicinal plants. For example, Sirih can help in indigestion and treat diabetes [7]. Mexican Mint is very good at solving skin problems, healing ulcers, and reducing diarrhoea and fever [11]. Likewise, Senduduk can treat skin problems and accelerate wound recovery [10]. Rerama and Belalai Gajah contain anti-cancerous and anti-inflammatory elements [16].

Manual identification based on human senses is highly related to their limited experience [14]. Furthermore, the identification process becomes difficult, especially when it involved a large number of species. It is necessary to preserve the information of these plants so that future generations can benefit from this knowledge [6].

Motivated in part by the importance of preserving the plants information in this work, image processing is used to support preserving the information. The advancement of image processing technology has improved due to the emergence of digital images, computer vision, object detection and recognition systems. For plant recognition, their parts such as leaves, flowers, roots and fruits can be used. In general, the shape of leaves is an essential and valuable characteristic for species identification. The leaf images need to go through the segmentation and features extraction stages to capture the geometrical information.

This work is an enhancement of Halim et al. [6] which focused only on seven (7) geometrical features extraction of herb leaves on the CPU. They have a shortcoming in producing accurate results since limited numbers of data can be considered. Therefore, this work tends to design an architecture to support the previous work by performing several calculations on the GPU.

The Graphical Processing Unit (GPU) was first introduced as a graphic card for gaming purposes. Recently, the use of GPU has been expanded to assist high computing. The use of GPU in computing has been applied in many areas, for example, in reconstructing 3D images from cloud points [5] and in 3D ocean modeling [2].

A GPU consists of thousands of threads (known as cores) that can perform various tasks simultaneously. The threads are grouped in several blocks, which form grids. The GPU is illustrated in Figure 1.

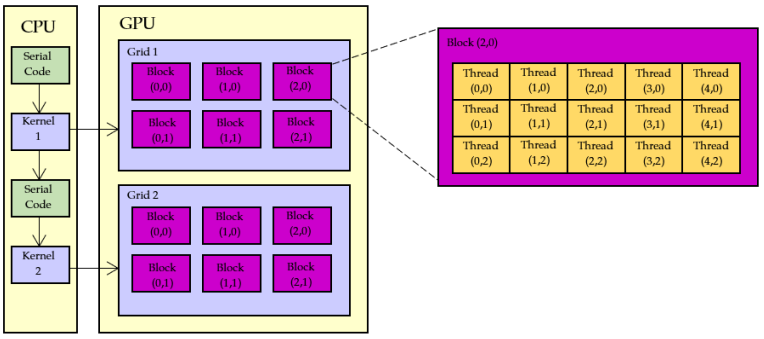

Figure 1: The GPU illustration.

Based on Figure 1, the entire general process is done sequentially in the CPU. However, at least one of the processes can be transferred to the GPU to be executed in parallel. The launching of the kernel on GPU is performed in parallel using data streams by a scalable array of multithreaded Streaming Multiprocessors (SMs) [1]. The number of kernels depends on the need of the whole process. Some existing work use only one kernel, for example, in denoising 3D point cloud [5], and some use several kernels such as in extracting surface points [4].

The design depends on the stream processing architecture that is suitable for computing massive parallel tasks [2]. This work used Compute Unified Device Architecture (CUDA) as its parallel programming platform.

This article is arranged as follows. Section 2 summarizes the related studies on plant segmentation and GPU usage. The methodology used in segmentation and features extraction is elaborated in section 3. Then, the processes are transferred to the GPU and discussed in section 4. Section 5 presents the results and the performance of the segmentation and the developed CPU-GPU platform. Finally, this paper ends with the conclusion in section 6.

2 Related Works

For leaf segmentation and features extraction on the parallel platform, limited previous works are found. [9] compared the performance between two GPUs (GeForce and Tesla). The process is split into two (2) and three (3) kernels, and each GPU computes the pre-processing such as colour conversion and the Sobel process. Although [9] use multiple kernels, all the kernels execute the same task. The two GPUs are run one after another. The results show that the 2-kernel give better speedup than 3-kernel, since the 3-kernel has increased the communication time between the processors. The developed architecture is tested on a Thunderbird image. Similarly, [3] uses a single GPU (GeForce) to compute the whole Sobel process on a grayscale image. The authors created only one kernel for the entire Sobel process. Sriramakrishnan et al. [13] also studied the performance of edge detectors on a single GPU (Quadro). The architecture is tested on 120 Magnetic Resonance brain tumor images (MRI) with the size of 240×240 pixels each. The author compares several edge detectors such as Sobel, Canny and Prewitt and found that Sobel gives the best speedup. This is because Sobel involves simple calculation compared to other edge detectors.

In addition to reducing the computational cost, the advantage of multi-GPU is that doing so can increase the size of the dataset. However, the architecture must be carefully designed to minimize the communication between CPU-GPU and GPU-GPU. [15] suggest two strategies for multi-GPU architecture called 'butterfly synchronization' and 'lazy update'. Pryor et al. [12] also employ multi-GPU to scan transmission electron microscopy, where multicore processors were utilized on CPU and multi-GPU architecture. [8] employs multi-GPU with the same specifications to do the training for cancer classification. Their work analyzes GPU memory which is not a common analysis in literature.

The previous literature shows that image segmentation and features extraction on the GPU platform needs further investigation. Therefore, this work has developed an architecture for both processes on a CPU-GPU platform with two GPUs. The data can contribute to the herb database, especially the Malaysian herbs, and can be extended to the classification of herb leaves. Furthermore, the proposed architecture will accelerate the whole process and increase the accuracy since a more extensive dataset can be considered.

3 Methodology

The general methodology of herb leaves features extraction is given in Figure 2.

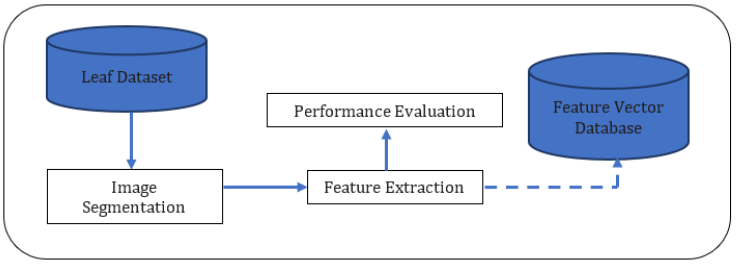

Figure 2: Herb leaves features extraction process flow.

Figure 2 shows the processes involved in processing the herb leaves. The input of this process is leaf images from the leaf's dataset. These images are segmented to detect the leaf's boundary. After that, the features extraction process is conducted. Finally, the performance of the developed architecture is assessed. The dashed arrow shows that the methodology can be extended as a feeder to the database development, which is not discussed in this work.

3.1 Leaf Dataset

Five (5) different herb leaves, Sirih (Piper betle), Mexican Mint (Plectranthus amboinicus), Rerama (Christia vespertilionis), Belalai Gajah (Clinacanthus nutans) and Senduduk (Melastoma malabathricum), are chosen as the data. Images of herb leave used in this work are collected and captured using Sony Alpha A6000 mirrorless digital camera with 24MP. All image resolutions are 1616×1080 pixels. A total of 125 images from five (5) types are randomly selected are used to test the effectiveness of the algorithm.

3.2 Image Segmentation

Sobel edge detection is used as image segmentation to detect the edges of the leaf. The approximate gradient of image intensity is computed using 3×3 kernels on horizontal and vertical changes as follows,

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}. \tag{1}$$

The gradient function for the leaf image $I(x, y)$ is given as follow,

$$g(x,y) = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \\ \frac{\partial I}{\partial y} \end{bmatrix} = \begin{bmatrix} G_x * I \\ G_y * I \end{bmatrix}, \tag{2}$$

where the G_x is the gradient changes in the horizontal direction, G_y is the gradient changes in the vertical direction. When the G_x and G_y are combined, it produces the gradient magnitude as in (3) and the gradient direction as in (4).

$$Magnitude\,|g(x,y)| = \sqrt{G_x^2 + G_y^2}, \tag{3}$$

$$Direction, \theta(x,y) = arctan\left[\frac{G_y}{G_x}\right]. \tag{4}$$

The eight (8) directional gradients of herb leaves are calculated as explained in [6]. From this process, binary images are obtained with the leaf edge.

3.3 Geometrical Features Extraction

This work utilizes seven (7) features to recognized herb leaves. The geometrical features are area, perimeter, major axis length, minor axis length, roundness, slimness, and smoothness [6] as provided in (5) to (11). Each of these features is useful for further investigation on the recognition and classification of leaf images.

$$area = \frac{1}{2}\sum(x_i y_{i+1} - x_{i+1} y_i), \tag{5}$$

$$perimeter = \sum \|p_i - p_{i+1}\|, \tag{6}$$

$$major = y_{max} - y_{min}, \tag{7}$$

$$minor = x_{max} - x_{min}, \tag{8}$$

$$roundness = \frac{4\pi \cdot area}{perimeter^2}, \tag{9}$$

$$slimness = \frac{major}{minor}, \tag{10}$$

$$flatness = \frac{1}{1+N}, \tag{11}$$

where (x_i, y_i) is the pixel of the i^{th} coordinate of the boundary point, p_i, the y_{min} and y_{max} is the lowest and the highest point of the vertical edge of the object's bounding box respectively, the x_{min} and x_{max} is the lowest and the highest point of the horizontal edge of the object's bounding box respectively, and N is the total number of boundary points.

3.4 Parallel Architecture

Image segmentation and features extraction processes discussed previously involved a large amount of data. Therefore, both processes are assigned to the GPU kernels to increase the computation performance. The GPUs are Tesla K20c and Quadro K4000, which have been installed in the CPU computer. This work uses MATLAB software to launch the kernels and to call the GPUs. The CPU computation is carried out on Intel (R) XEON (R) (2.10GHz) with two (2) processors. The detail of the CPU-GPU implementation is discussed in the next section.

4 CPU-GPU Implementation

The entire process in Figure 2 requires high processing time due to a large number of leaf datasets. Hence, image segmentation and features extraction are chosen to be executed on the GPU since both processes require high execution time in a single CPU. Two (2) GPUs known as GPU 1 and GPU 2 are employed in the process to perform (2) to (4), respectively. This compares each GPU's ability to complete the task and fully utilize the existing GPUs in the system. The process of data passing from CPU to GPU is illustrated in Figure 3.

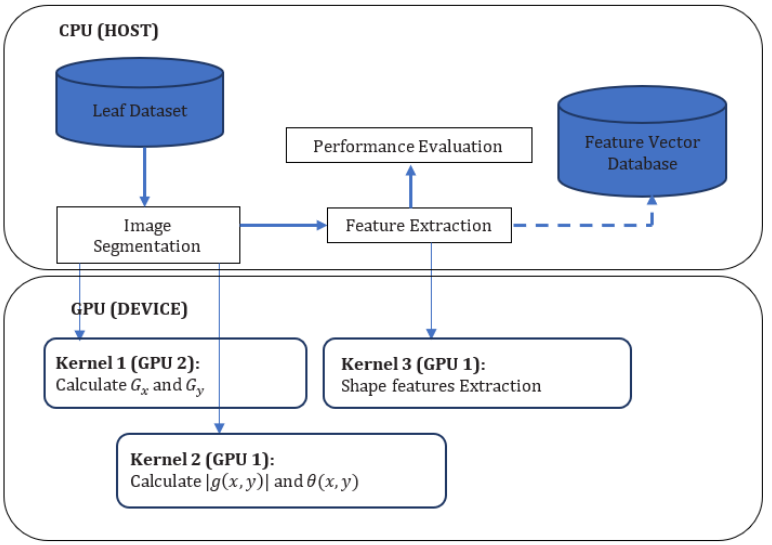

Figure 3: The CPU-GPU architecture for herb leaves features extraction.

Based on the figure, after Kernel 1 has completed the calculation of G_x and G_y, it will pass the data to the CPU to be transferred to Kernel 2 for $|g(x,y)|$ and $\theta(x,y)$ calculation. The reason for choosing GPU 2 for Kernel 1 and GPU 1 for Kernel 2 is described in section 5.2. The feature extraction process needs only one GPU since it involves uncomplicated calculations. It is executed on GPU 1 since GPU 1 is the default GPU in the machine.

5 Results and Discussion

This section presents the results of the extracted boundary of leaf image using Sobel and analyses the performance of the CPU-GPU architecture.

5.1 Geometrical Features Extraction

Figure 4 shows the results of the original image, segmented image, and the extracted boundary for each sample of herb leaf species.

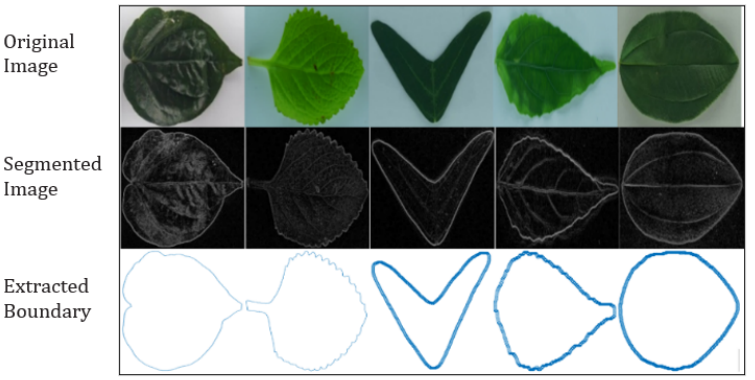

Figure 4: Sample of Malaysian herb leaves results. Left to right: Sirih, Mexican mint, rerama, belalai gajah, senduduk.

Although the Sobel edge detector is sensitive to noise, in this data, the usage of the Sobel edge detector is sufficient to produce a clear edge of the image leaf. The blurred edges for Sirih and Mexican are due to the selection of threshold during the detection process.

The performance of segmented images and extracted features will not be examined. This work focuses on CPU-GPU architecture to accelerate the computation time, which will not be associated with the accuracy of the method. However, the CPU-GPU architecture is essential to ensure that more images can be considered to boost the reliability of the database.

5.2 CPU-GPU Performance

The computational performance of the designed CPU-GPU architecture in Figure 3 is assessed in terms of processing time, speedup and efficiency.

5.2.1 Processing Time and Speedup

Processing time is the cumulative time taken by the machine to complete a task. The processing time is used to calculate the speedup, S as,

$$S = \frac{T_{seq}}{T_{par}}, \tag{12}$$

where T_{seq} is the processing time for the sequential process on CPU and T_{par} is the processing time for the parallel process [1].

This work measures two (2) different speedups. The first speedup is between two (2) different architectures which are GPU1-GPU2 and GPU2-GPU1. The second speedup is measured only for Kernel 3 with GPU 1 by changing the image pixel values.

The first speedup involves two (2) kernels and two (2) GPUs as shown in Figure 3. Both kernels are tested on different GPUs simultaneously to check the capability of the GPU and to fully utilize the equipped GPUs. Kernels with the respective GPU architectures are shown in Table 1.

Table 1: Kernels for GPU 1 and GPU two (2) for different architectures.

Architecture	**GPU 1**	**GPU 2**
GPU1-GPU2	Kernel 1	Kernel 2
GPU2-GPU1	Kernel 2	Kernel 1

The first architecture is called GPU1-GPU2, where GPU 1 executes Kernel 1, and GPU 2 executes Kernel 2. Similarly, GPU 2-GPU1 architecture means that GPU 2 executes Kernel 1, and GPU 1 executes Kernel 2. Kernel 1 and Kernel 2 are evaluated as a process with two (2) GPUs, but the task for each GPU is different. Thus, the performance depends on the performance of the GPUs. The architectures are executed one after another, and the processing time is recorded as in Figure 5.

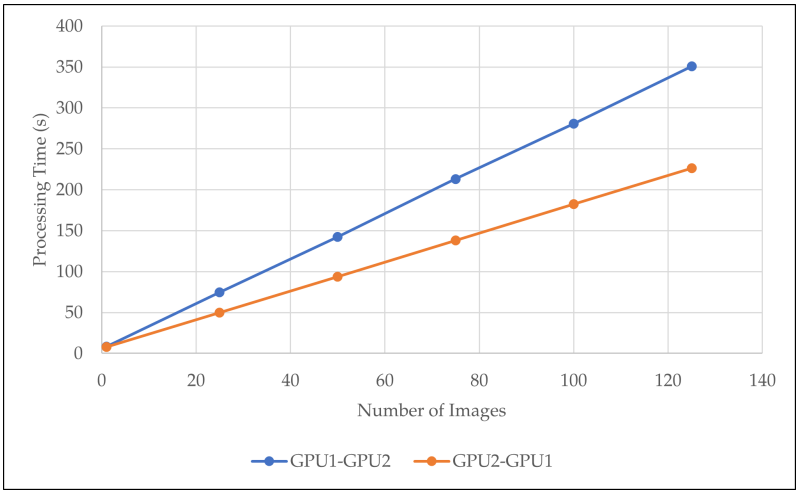

Figure 5: The processing time of two different architectures using two GPUs for different number of images for kernel 1 and kernel 2.

In Figure 5, x-axis shows the number of images per process, from 25 to 125 images. These images are processed concurrently. The highest processing time for GPU1-GPU2 is 350s and 226s

for GPU2-GPU1 for 125 images. From the figure, GPU1-GPU2 architecture consumes higher processing time compared to GPU2-GPU1 for all numbers of images. This comparison is further explained by comparing their speedup as given in Figure 6.

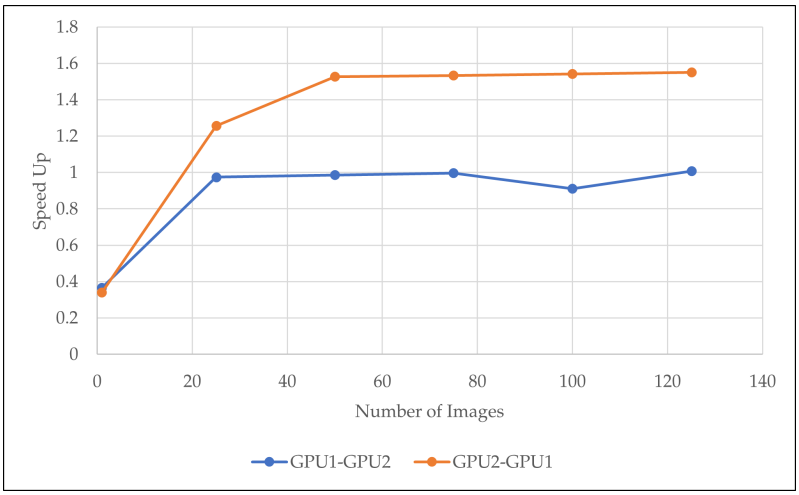

Figure 6: The speedup of two different architectures using two GPUs for different number of images for kernel 1 and kernel 2.

In Figure 6, the y-axis displays the speedup when the number of images and data points is increased, as shown in the x-axis. With GPU, the processing time has been accelerated. The GPU2-GPU1 architecture has a better speedup up to 1.55 factor compared to GPU1-GPU2 architecture with only 1.01 factor. This performance shows that Kernel 2 is more suitable to be executed on GPU 1. This is because Kernel 2 involves more complicated computation such as power and trigonometric as compared to Kernel 1 (refer to Figure 3 and equations (3) to (4)), which suit the compute compatibility of GPU 1.

The second speedup is measured only for Kernel 3 in GPU 1, involving the features extraction method. The computation is less complicated since the number of data points has been reduced to only extracted boundaries. Moreover, the mathematical calculation involved is less complicated (refer to (5) to (11)). Thus, only GPU 1 (Tesla K20c) is employed in the computation.

For comparison purposes, the number of pixels is reduced from 1616×1080 to 768×513 pixels to analyse the GPU performance in terms of processing time, speedup and efficiency. The CPU and GPU processing time is given in Figure 7.

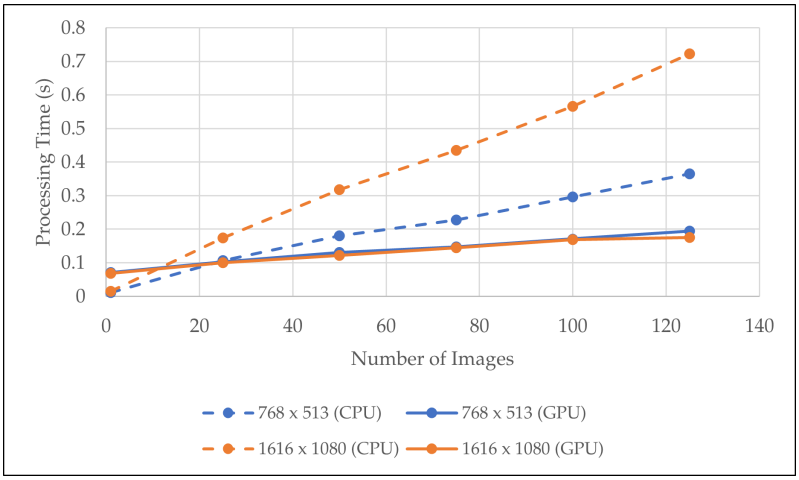

Figure 7: The CPU and GPU processing time for different number of image and image pixels for kernel 3.

In Figure 7, the CPU processing time is presented in dashed-line, and the GPU processing time is presented in solid-line. The blue lines are for 768×513 pixels, and the orange lines are for 1616×1080 pixels. The CPU time has been multiplied from 0.365s to 0.722s when the number of pixels is increased. This is because the CPU executes the pixels sequentially. On the contrary, the GPU processing time is better for 1616×1080 as compared to 768×513. It can be observed that the orange line has more time difference between the CPU and GPU compared to the blue line. This shows that the processing time has been improved better for 1616×1080. The GPU speedup is given in Figure 8.

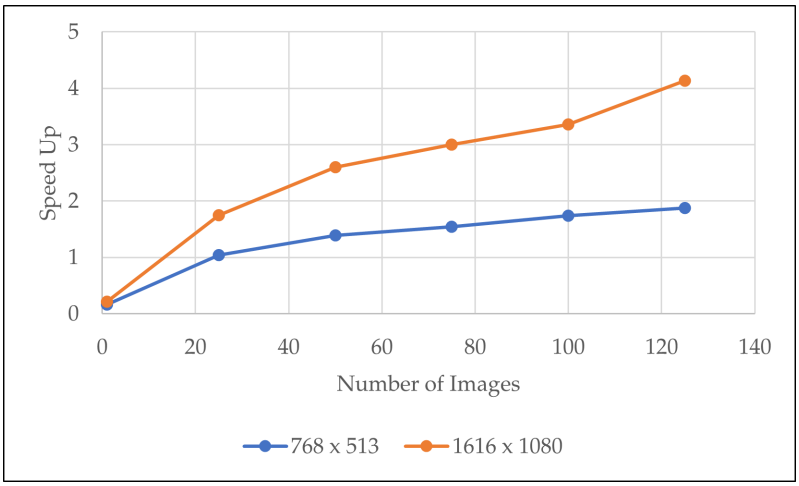

Figure 8: The GPU speedup for different number of image and image pixels for kernel 3.

In Figure 8, the y-axis shows the speedup when the number of images and data points is increased as in the x-axis. Based on the figure, the increment of pixel numbers in both cases has accelerated the feature extraction computation process by a factor of 1.87 and 4.13 for 768×513 and 1616×1080 pixels, respectively. The comparison between two (2) different sets of pixels shows that GPU gives better speedup for a higher number of data points. This is because the GPU has hundreds of threads that are ready to do the task. Underutilization of the threads may produce

idle time and increase the total execution time and decrease the speedup. The reduction of pixel numbers to help the CPU process the data is insignificant, consequently reducing the image quality. Figure 8 also suggests that it is unnecessary to loosen the quality of images if GPU is added to the computation.

5.2.2 Efficiency

Besides speedup, this work measures the parallel performance by parallel efficiency. The efficiency value shows the processors utilization and the average contribution of the processors towards the global computation. The efficiency, E is calculated as in (13).

$$E = \frac{S}{N_{par}}. \tag{13}$$

S is the speedup calculated in (12) and N_{par} is the number of processors used in the process [1]. In this experiment, the number of processors is the number of threads per block. The efficiency is estimated for all three (3) kernels in Figure 3 and illustrated in the following figures.

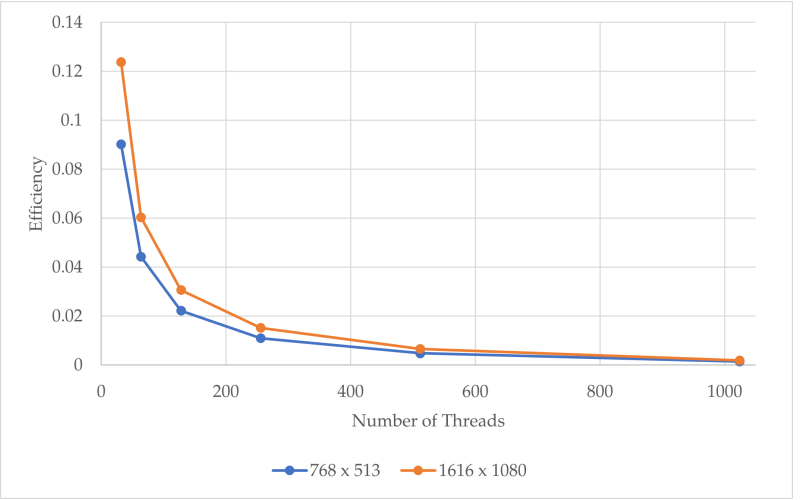

Figure 9: (a) The efficiencies of kernel 1.

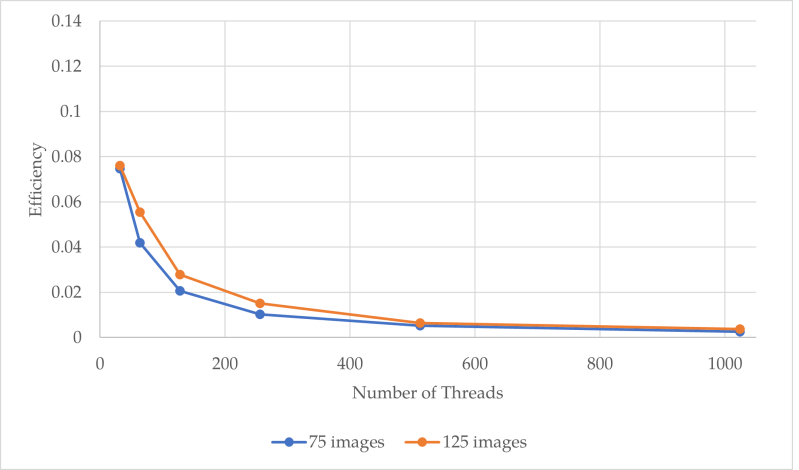

Figure 10: (b) The efficiencies of kernel 2.

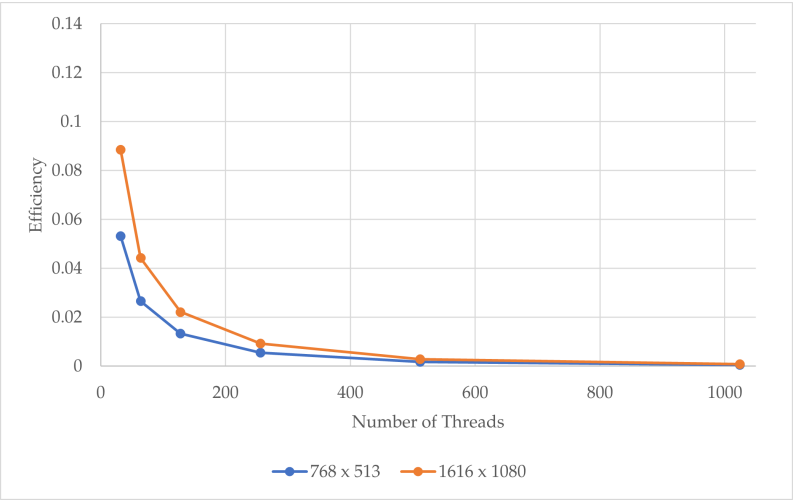

Figure 11: (a) The efficiencies of kernel 3.

The figures show that the efficiencies for all three (3) kernels. For Kernel 1 and 2, the efficiencies are measured for 125 images with two (2) different pixel values: 768×513 and 1616×1080. For Kernel 3, since it does not comprise pixel values, the efficiency is compared for a different number of images: 75 and 125 images. As happened to speedup, the efficiencies for a larger amount of data are better than the lower one. However, all kernels have lost the efficiencies with the increment of processors number. This loss may occur due to some circumstances such as processors communication, data transmission, and task partitioning [1]. The efficiency can also degrade if the number of processors is more and the amount of computations.

The figure shows that the best efficiency for all kernels is when the number of threads is the smallest, in this case, 32 threads. This is because each thread is fully occupied with the task. Nonetheless, when the number of threads approaches the maximum, 1024, the efficiencies are degraded, showing the excessive number of threads.

6 Conclusions

This work presents an architecture to executes herb leaves images processing on a parallel platform. The experiment chooses images of Sirih Mexican Mint, Rerama, Belalai Gajah, and Senduduk for the analysis since these are among the popular herbs used in traditional medication. Sobel edge detection is adopted as a segmentation method for all leaf images. Then, the features are obtained from the generated extracted boundary points. Seven (7) geometrical features are considered in this work: area, perimeter, major axis length, minor axis length, roundness, smoothness, and flatness calculated from each image. Sobel edge detection provides good segmentation results, increasing the correctness of the extracted features.

Original image with 1616×1080 pixel dimensions each requires a lot of computation time to be analyzed. Therefore, the computation time for segmentation and features extraction processes is accelerated by executing the operations on the CPU-GPU platform. Since there are two (2) GPUs with different capabilities occupied in this platform, the comparison between the two is also made by assigning different tasks in the features extraction process. The first comparison is called GPU1-GPU2, to handle Kernel 1 and Kernel 2, respectively. Then, the role of these GPUs is switched. From the result, the GPU2-GPU1 platform gives a better speedup since Kernel 2 is more suitable for executing on GPU 1 due to the higher capability of GPU 1.

The second comparison is on the features extraction process where only GPU 1 is utilized. The analysis is made on two (2) different pixels dimensions: 768×513 and 1616×1080. The speedup has been improved by increasing the data size: the number of images and the pixel dimensions. This is because the processors have been fully utilized without waiting or idle time. Furthermore, the speedup also recommends that more images be considered in the whole process, improving the accuracy of the related process such as detection, recognition, and classification. Good speedup is essential to ensure that the output can be produced in real-time.

Besides speedup, the performance of the architecture has been measured using efficiency. The value reflects the processors utilization. The results show that all three (3) kernels have degrading efficiencies when the number of threads increases. This phenomenon happens when the load balance is not achieved. Consequently, the architecture needs to be revised to improve efficiency.

In the future, it is recommended to include other types of features such as textual, colour and shape features to capture more relevant information. The extracted features from this work can as the input of further processes as mentioned above. Consequently, this work supports developing the Malaysian herb leaves database as a source for bioproduct and eco-tourism for Malaysia.

Acknowledgement This research is supported by FRGS-RACER (600-IRMI/FRGS-RACER 5/3 (053/2019) and RACER/1/2019/STG06/UITM//4). This research is a collaborative research between Universiti Teknologi MARA and Universiti Teknologi Malaysia.

Conflicts of Interest The authors declare no conflict of interest.

References

[1] A. Afzal, Z. Ansari & M. K. Ramis (2020). Parallel performance analysis of coupled heat and fluid flow in parallel plate channel using CUDA. *Computational and Applied Mathematics*, *39*(3), 1–25. https://doi.org/10.1007/s40314-020-01244-1.

[2] A. A. Dahawi, N. B. Alias & A. Idris (2021). GPU parallelization for accelerating 3D primitive equations of ocean modeling. In *Advances on Smart and Soft Computing*, pp. 643–654. Springer, Singapore. https://doi.org/10.1007/978-981-15-6048-4_56.

[3] H. B. Fredj, M. Ltaif, A. Ammar & C. Souani (2017). Parallel implementation of Sobel filter using CUDA. In *2017 International Conference on Control, Automation and Diagnosis (ICCAD)*, pp. 209–212. IEEE, Hammamet, Tunisia. https://doi.org/10.1109/CADIAG.2017.8075658.

[4] N. A. Hadi (2019). Big data simulation for surface reconstruction on CPU-GPU platform. In *Journal of Physics: Conference Series*, pp. 1–8. IOP Publishing, Bandung, Indonesia. https://doi.org/10.1088/1742-6596/1192/1/012006.

[5] N. A. Hadi, S. A. Halim & N. Alias (2021). Statistical filtering on 3D cloud data points on the CPU-GPU platform. In *Journal of Physics: Conference Series*, pp. Article ID: 012006. IOP Publishing, Chennai, India.

[6] S. A. Halim, N. A. Hadi & N. S. M. Lazim (2021). Segmentation and features extraction of Malaysian herbs leaves. In *Journal of Physics: Conference Series*, pp. Article ID: 012005. IOP Publishing, Chennai, India.

[7] S. Handoo (2021). *What are the benefits and side effects of betel leaves?* https://www.stylecraze.com/articles/medicinal-uses-of-betel-leaf/#gref.

[8] T. Haryanto, A. M. Arymurthy, H. Suhartanto & K. Kusmardi (2020). GPUs utilization of residual network training for colon histopathological images classification. In *2020 International Conference on Computer Science and Its Application in Agriculture (ICOSICA)*, pp. 1–8. IEEE, Bogor, Indonesia. https://doi.org/10.1109/ICOSICA49951.2020.9243276.

[9] A. Jain, A. Namdev & M. Chawla (2016). Parallel edge detection by SOBEL algorithm using CUDA C. In *2016 IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, pp. 1–6. IEEE, Bhopal, India. https://doi.org/10.1109/SCEECS.2016.7509360.

[10] H. Napisah, A. Azmahani, A. L. Zubaidi, A. Intan & A. Nazifah (2011). A preliminary study on the antimicrobial properties of several plants collected from Terengganu, Malaysia. *Journal of Agrobiotechnology*, *2*, 99–106.

[11] S. L. Ong, S. Paneerchelvan, H. Y. Lai & N. K. Rao (2014). In vitro lipase inhibitory effect of thirty two selected plants in Malaysia. *Asian Journal of Pharmaceutical and Clinical Research*, *7*(2), 19–24.

[12] A. Pryor, C. Ophus & J. Miao (2017). A streaming multi-GPU implementation of image simulation algorithms for scanning transmission electron microscopy. *Advanced Structural and Chemical Imaging*, *3*(1), 1–14. https://doi.org/10.1186/s40679-017-0048-z.

[13] P. Sriramakrishnan, T. Kalaiselvi & K. Somasundaram (2018). Parallel processing edge detection methods for MR imagery volumes using CUDA enabled GPU machine. *International Journal of Computer Sciences and Engineering*, *6*(4), 123–130.

[14] C. L. Suan, N. Alias, R. Januari, M. N. Mustaffa, A. Ali, M. H. A. Kamal & I. Hayat (2016). Mathematical modeling for contour identification based on medicinal leaves and gis images. *Jurnal Teknologi*, *78*(49–55). https://doi.org/10.11113/jt.v78.10142.

[15] R. Wu, S. Yan, Y. Shan, Q. Dang & G. Sun (2015). Deep image: Scaling up image recognition. *arXiv preprint arXiv:1501.02876*, *7*(8).

[16] A. Zakaria (2015). *UPM runs stage two of anti-cancerous red butterfly wing research*. https://www.upm.edu.my/news/upm_runs_stage_two_of_anti_cancerous_red_butterfly_wing_research-25072.