Phase 1 Report Literature Review

Antrea Christou & Calvin Greenewald

8 February 2024

11. Comparison of the Efficiency of Parallel Algorithms KNN and NLM Based on CUDA for Large Image Processing

Lesia Mochurad, Roman Bliakhar

The purpose of this paper is to address the problem of noise reduction in large satellite images and use parallel processes alongside the KNN and NLM algorithms to reduce noise. The authors outline how easily images can be contaminated with noise, and that this extra noise may prevent models from accurately processing and analyzing an image. In all areas of image processing, accurate processing and segmentation is very important to train a model correctly and get correct classifications of the images. The addition of noise is a limiting factor which can throw off the processing and analysis of an image.

The authors are trying to tackle this project to help researchers and companies alike who use satellite images. They are trying to ensure that models can accurately process images without losing important information due to the added noise. Their method attempts to reverse the effects of the added noise and present the image with minimal "loss of original features [1]," and with minimal to no noise.

This paper uses the Non-Local Means (NLM) and k-Nearest Neighbors (KNN) algorithms to reduce noise. The principle behind NLM is that the algorithm reduces noise by finding other pixels in the image that are similar to the noisy pixel that is getting replaced, and the values of all the other similar pixels are averaged and the noisy pixel is replaced with this average value. The NLM algorithm proves to be more effective at reducing noise and have "less loss of image detail compared to average local algorithms [1]." The KNN filtering algorithm defines a pixel by being influenced by its surrounding pixels. Therefore, the similarity between a pixel and its k-neighbors are calculated and then weights are assigned to the pixels based on their

relationship to the target pixel. The target pixel is then updated with respect to its surrounding pixels and their assigned weights. Each of the NLM and KNN algorithms are then run on CUDA to help with the parallel processing of each algorithm. They also change thread counts to determine if the number of running threads has an impact on the execution time and accuracy of noise reduction.

The results of this paper show that both NLM and KNN are effective at performing noise reduction in images. There are however subtle differences in image clarity, but both reduce noise well and keep the original characteristics of the images. Most importantly however, the use of CUDA and parallel processing was far superior to simply using a CPU as the processing times were significantly cut with the use of CUDA. Additionally, results show that more threads used in CUDA directly correlates to a decreased processing time. Some results for the use of 1024 threads show that processing times for CUDA are up to forty times less than the processing times for the CPU. These results outline that the use of parallel processing in image processing can be very beneficial as it cuts the processing time down significantly. This ensures that parallel processing techniques can be used to process large datasets of images, or even better process images in real time.

12. Parallel Image Segmentation using Multi-Threading and K-Means Algorithm

Soumyo Bose1, Aniruddha Mukherjee1, Madhulika1, Sayan Chakraborty1, Sourav Samanta, Nilanjan Dey

Image segmentation is an important application of computer vision and machine learning. This method breaks images into segments and then can classify objects or edges in each of the segments. This process can be time intensive, but this time can be decreased with the use of multithreading techniques. This research paper aims at decreasing the segmentation time through the use of multithreading.

Reducing the time of image processing and segmentation is an important project as these algorithms can be extremely time intensive. This is a limitation especially in scenarios where time is of the essence, like real time segmentation and processing. The researchers are attempting to decrease the time of image segmentation to help process and classify images more quickly.

This project takes the images and breaks them into 2, 4, or 6 parts, each of which be passed to separate threads to be run in parallel. These multiple parts are then segmented using the k-means clustering algorithm. This algorithm works by randomly assigning k clusters and then assigning a center to those clusters. The other pixels in the image are then put in the clusters which are closet to their pixel values. The clusters are then averages and this average value is the centroid for the next cluster assignment. This process continues until the centers do not change significantly. This research paper employs the use of k-means by taking the different parts of the image and running the k-means algorithm on them. These different parts are run in parallel until the parts of the image have successfully been segmented by the k-means algorithm. The parts of the image are then rejoined for the full image. The use of parallel processing in this case

significantly decreases the time required for segmenting the image through the use of the k-means algorithm.

The results of this project showed that the researcher's method of incorporating multi thread processes into their method of image segmentation greatly reduced the execution time of image segmentation. Furthermore, their method also maintained accurate and reliable results while cutting down the execution time considerably from traditional non-parallel techniques. The image segmentation was carried out on multiple images of different sizes and the images were processed using a different number of threads. The experiment covered images with 1, 2, 4, and 6 threads. The results showed that the processing time for the images with multiple threads was, in some cases, less than half that of the time that it took to process the image with 1 thread. With the lower clusters, the total processing time across the different number of threads did not change much, but as the clusters increased, the processing time decreased the more threads there were.

The researchers discuss potentially expanding their project to try their method on different datasets or images. Additionally, they expressed interest in further expanding their method which could then be used in a number of different cases.

13. Real Time Digital Image Processing Using Point Operations in Multithreaded Systems

Samyan Q. W1, Sahar W.2, Talha W.3, Aslam M

In this paper, the researchers are trying to tackle the common problem of speeding up image processing. Image processing can be computationally expensive and require state of the art processors and memory for the image to get processed quickly and accurately. The problem is especially prevalent in the processing of real time images or videos such as security footage or autonomous car video feeds. While there are other researchers out there taking on the same problem, according to this paper, many of them take on the problem in one of 2 ways, class A or class B. Class A research is centered around manipulating the images to give the processer a better image. This includes adjusting brightness, contrast, or other image enhancers. Class B research is targeted at dividing the images into other smaller images and processing each smaller image on a different thread. This is a problem because the relationships between pixels can be lost when they are separated into the smaller images.

Therefore, the researchers of this paper are trying to speed up image processing while maintaining accuracy and the quality of the image. Processing the whole image together is important to not lose the relationships between pixels. This solution attempts to process images more efficiently and more accurately.

This research paper uses a technique which combines class A and class B. To begin, they divide the image into a certain number of equal smaller images. The threads then do some enhancement on the smaller images before then putting them into a multithread's convolution. The convolution works by moving a smaller matrix across the entire image and getting the dot product of each matrix and then sticking those values into another matrix. What this does is it give the model better knowledge of how all the pixel values relate to each other which then can

help the model understand edges Because the image was broken up into smaller images, the model can’t see the other pixels that used to be adjacent to it in the original image, so the smaller matrices are put into vertical threads and then the threads are allowed to communicate with each other so that each thread can learn from the other pixels in the convolution stage.

The results show that a certain increase of threads leads to a quicker execution and accurate results, but after that certain level, the performance and execution time decreases.

Due to the positive results of this research, further research could be done on this topic. The researchers could next use these techniques to tackle the problem of real time video processing. This could be useful for security videos or could be incorporated into autonomous vehicles. The positive results of this research offer a good chance that there will further research done on this topic. Additionally, the researchers would like to test their work on different operating systems to determine if other systems may have larger and more significant impacts on the processing time of images and real time videos.

14. Multithreading Image Processing in Single-core and Multi-core CPU using Java

Alda Kika and Silvana Greca

This problem research paper is aimed at tackling the problem of image processing through Java's framework. Java supports multithread programming and therefore makes it easier for developers to use multithread algorithms.

Th researchers are trying to determine the most efficient way of multithread image processing by testing different images and different complexities of algorithms, as well as testing on single vs multi core processors. By complexity, they mean the number of calculations that the thread needs to do to accurately process the image. In this research, the researchers gave the algorithm 3 different kinds of images. The first changed the brightness, the second, changed the contrast, and the third included a steganographic message.

This research experimented with both single-core and multi-core processing, and it was determined that the multithreading algorithm was beneficial to improving the overall performance and speed of the model. Although the results were different based on the core, the multithreading approach had a positive impact on the processing of images. Single-core processors performed best with smaller images and less complex algorithms, whereas the multi-core processors performed best with smaller images and more complex algorithms. This goes to show that multithreading techniques are capable of increasing the performance of more complex image processing algorithms on multicore systems. The results also show that an increase in the number of threads leads to a faster execution and more accurate results but only up to a certain point. Beyond this, the performance and execution time start to decrease. This suggests that there is a best number of threads for achieving the best performance when using this research team's method.

The processing time for the algorithms increased as the number of threads increased until the number of threads surpassed 10. After which, the processing time plateaued and somewhat

increased. In the single-core process, the results were similar in that after a certain number of threads the processing time started to increase.

The researchers stated that in future work, they would like to dive deeper into the process and algorithm to determine the cause of the results that they got. This could allow them to make any necessary changes to get more trustworthy results. Additionally, they determined that they would like to try their method in different languages and on different operating systems to determine if that has an influence on the results at all.

15. Distributed Parallel Image Signal Extrapolation Framework using Message Passing Interface

Jurgen Seiler and Andre Kaup

As image and video technology continues to improve, the quality and resolution of images and videos has significantly increased as well. This increase in quality has a negative impact on processors, however. The increased quality requires more computational power and more advanced processing algorithms. This research paper uses image signal extrapolation, a set of processing algorithms. While they play an important role in many image processing tasks, they are computationally demanding. These algorithms work by extrapolating parts of the image onto other parts of the image that are missing. As the images and videos continue to get better and better, requiring further computational power for processing, the extrapolation algorithms require updates as well to allow for faster processing.

More efficient algorithms are paramount for the success of these more complex image processing requirements. For this research, the Frequency Selective Extrapolation algorithm was used. The Frequency Selective Extrapolation works by dividing the image into multiple sections and then using the knowledge in the surrounding pixels, and pixels in other image sections to generate filler pixels to go in the place of missing pixels in the image.

To solve their parallel image signal extrapolation method, they sectioned each core on the CPU to take a specific part of the image. They used an 8 core CPU which gives 32 processing nodes. The parallelization is done with an OpenML algorithm. The images are then broken up into multiple sections and each section is assigned a processing core. The sections of the image slightly overlap other sections as to not lose information. Also, this allows the processors to gather enough information from the entire image without being limited from being divided into multiple sections.

The results of this research conclude that while keeping the original image signal extrapolation algorithms and simply parallelizing them through the process of dividing the

images into smaller sections, the image processing time is greatly reduced. In fact, when there were 32 nodes, the processing time was reduced by a “factor of 22 [page 5].” Most importantly, in addition to the parallelizing technique significantly reducing the processing time, the accuracy of the processing was not changed. What this means is that the image processing algorithms had essentially the same results when processing the whole image, verses processing the image parallelly, while the parallel processing decreased the time significantly. Further research could be directed at making other signal processing algorithms more efficient.

16. PARALLELIZATION OF VIDEO PROCESSING: FROM PROGRAMMING MODELS TO APPLICATIONS

Dennis Lin, Xiaohuang Huang, Quang Nguyen, Joshua Blackburn, Chris Rodrigues,

Thomas Huang, Minh Do, Sanjay Patel, and Wen-Mei Hwu

The significant increase in the use of video technology, from security videos to autonomous vehicles, has increased the need for advancements in video processing techniques and algorithms. Additionally, videos are becoming more difficult to process due to their increased resolution and size. These advancements in the quality of videos call for similar advancements in processing technology that can be done in real time on the video feed.

This is an important problem to solve as real time video processing would allow for immediate processing of security footage and other videos. Real time security footage processing could help determine unexpected behavior from people or things in the footage, thus allowing for quicker preventative measures. While these advancements may have been limited in the past due to computational constraints, chip companies are releasing advanced processing cores with "more than 10s of cores [page 2]." This is a significant increase of computational power and will allow for more complex video processing algorithms to be executed.

This research optimizes the CPU but also moves the more complex parts of the computations over to a GPU to save time. The performance of this method shows decreased processing times, up to 13 times less than simply using the CPU.

The image is spilt up for processing because especially in videos, one frame does not necessarily relate to another frame. Additionally, pixels do not always relate to or have an impact on other pixels even if they are adjacent. Therefore, the images are divided up and each processed separately by a core. In the event that there needs to be communication between processors because of related pixels getting separated, shared memory access is utilized. This

allows for each processor to communicate with other processors to gather necessary information about other sections of the image.

The results of this research paper show that the combination of optimized CPU processing and giving computationally expensive tasks to the GPU significantly enhances video processing capabilities. Using the advancements in processor technology allows for better advancements in video processing. The researcher's method of optimization is a promising advancement in the field as it showed a significant decrease in video processing times. This will help allow for real-time video processing for security footage or other video feeds.

17. Parallel Video Processing Techniques for Surveillance Applications

Leonidas Deligiannidis, Hamid R. Arabnia

This research paper targets the problem of real time security video processing. Video processing is a complex program which requires powerful computers to process accurately and quickly. The idea of parallelly processing surveillance videos “is to collect and disseminate real-time information” which can then be passed on to be analyzed by security officers or other analysts [page 1].

The purpose of this research is to improve the efficiency of surveillance cameras across many domains and improve the quality of processing. Improved processing accuracy and time could allow for quicker recognition and halting of unwanted activity happening in the area which the camera is trying to observe. Also, it could allow for analysis to be done in order to observe a past crime. Having advancements in parallel video processing could allow for improved speed of processing, which could potentially allow for real-time video processing. The parallel process is done with CUDA where each individual pixel can be processed by 1 thread in the processing algorithm.

This research presents multiple parallel algorithms which can be applied to the surveillance videos to better process and analyze them. The first is a motion detector which employs a GPU to compare the current frame with the previous frame. If there are a certain number of pixels which have significantly changed, motion is detected which can then raise an alarm [page 4]. The second is over the line motion detection. This is where motion is only detected in a target part of the videos [page 4]. The third is a “line crossing detector” where motion is detected “on the line [page 5].” The fourth and final algorithm is area motion detection

which can define a region to be searched for motion by defining multiple edge lines surrounding the target area.

These processing algorithms all work to improve the effectiveness of surveillance video processing. The implementation of parallel processing allows for the algorithms to speed up the processing of real time video feeds by employing CUDA.

18. Parallel processing for image and video processing: Issues and challenges

Alain Merigot, Alfredo Petrosino

The purpose of this research paper is to outline some issues developers face when implementing parallel processing videos and images. The paper discusses the history of trying to solve the problem of image processing, as well as the challenges faced which resulted from limitations in processing power.

Additionally, the writers stated that coding languages are large limitation for developers of parallel algorithms. When the code is transferred to machine readable assembly code, the assembly code is not written effectively or efficiently. This causes delays or negates the true potential of the parallel algorithm. These efficiency errors usually have to be corrected by hand by the authors of the algorithm [page 695].

The writers suggested that to bypass this problem, a coding language should be made to exclusively parallelize image and video processing algorithms. They mentioned that there are already some languages which make attempts at parallelizing each part of the image, but the writers suggested that these languages need to be updated.

The conclusion of this research paper outlined the need for advancements in processing technology. Currently, the processing chips used for video and image processing are too expensive and don't quite meet the requirements for processing power. There is an increasing demand for chips to have more processing power but less energy usage and a smaller overall size. These requirements are for affordability, and versatility. It is easier to put processing chips in smaller devices if the processing chips are a smaller design and don't require large energy

sources. Furthermore, parallel algorithms are required to speed up the processing speed of videos. With computational limitations, to process videos traditionally, takes far too long.

19. Parallel Image Segmentation Using Reduction-Sweeps On Multicore Processors and GPUs

Image segmentation is a popular form of dividing images up into smaller sections which can each be processed simultaneously in parallel. Processing smaller segments individually can lead to a loss of information. While there are techniques used to minimize this loss of information, they would still be processed more accurately if they were processed sequentially [139]. This loss of information could lead to misclassifications, errors in detection, or other serious faults. To mitigate this, the researchers then introduce their contribution called graph-based image segmentation.

Their approach is targeted at better categorizing the pixels in the images, by graphically dividing the image. Each pixel is originally its own section but then pixels and groups of pixels are added to sets that they are similar to. At the end of the segmentation, there are groups of similar pixels on the graph. The similar pixels are then joined together, and the average is taken. The values are then smoothed and according to a gaussian curve and the pixels are then reordered back into the image.

The results from this research paper showed that the decreased time from parallelizing the processing was not as much as they expected, and therefore the algorithm did not perform

well enough. Furthermore, the writers expressed that often times the parallel algorithm actually took longer than the sequential algorithm.

The researchers outlined some possible future work as adding features to ensure that the memory remained in the memory of the CPU rather than getting mover around every time it is used. This would allow for the algorithm to process quicker and without using up as much space. Additionally, they stated that increasing the segmentation of the images “from 2D to 3D” would be another area for future research [145].

20. Image Processing Tasks using Parallel Computing in Multi core Architecture and its Applications in Medical Imaging

Sanjay Saxena, Neeraj Sharma, Shiru Sharma

The problem presented in this paper is in regard to the need for medical imaging to be processed more efficiently and quickly. Delayed processing could result in errors in diagnosis, prolonged hospital visits as the patients await results, and or other inconveniences. Medical imaging can require large amounts of computational power, and it also requires accurate observations from the processor. Medical image processing cannot afford to be inaccurate.

The purpose of the research is to parallelize medical image processors to speed up the processing time while still getting accurate results. Parallel image processing can be extremely beneficial to the medical field as parallel algorithms can be used enhance quality, and other image processing techniques quickly and accurately. The increased speed of processing allows for faster diagnosis, and medical response to ailments.

Their method was to parallelize many commonly sequential image enhancement and segmentation algorithms. These include, “Parallel Segmentation by Region Growing” which highlights regions of similar pixels [page 3]. Additionally, another algorithm was “Parallel Segmentation by Global Thresholding” which is done by calculating thresholds and then assigning pixels to those thresholds. Another parallel technique the researchers implemented was “Noise Reduction [page 3].” This enhances the quality of the image by sharpening the pixels. Finally, the last parallel technique the researchers implemented was “Histogram Equalization

[page 3].” This technique results in a normalized set of pixels in a histogram format which is then processed [page 3].

The results of this research showed promising results in the processing time for images. The researchers tested image enhancement as well as segmentation and the results showed that the parallel algorithm is up to 2.5 times faster than a traditional sequential algorithm. This is a large boost especially in the field of medical imaging where time is an important factor.

The researchers proposed that in future work they would like to negate the requirement of large memory space, especially for large images, and create an “out-of-core” algorithm that would allow for additional segmentation and processing of large images that are too big for the memory space [page 5].

References

[11] L. Mochurad and R. Bliakhar, “Comparison of the Efficiency of Parallel Algorithms KNN and NLM Based on CUDA for Large Image Processing,” Computer Modeling and Intelligent Systems, vol. 3137, pp. 238–249, 2022, doi: https://doi.org/10.32782/cmis/3137-20.

[12] S. Bose, A. Mukherjee, S. Chakraborty, S. Samanta, and N. Dey, “Parallel Image Segmentation using Multi-Threading and K-Means Algorithm,” ResearchGate, Dec. 2013. https://www.researchgate.net/publication/267329073_Parallel_image_segmentation_using_multi-threading_and_k-means_algorithm.

[13] Q. W. Samyan, W. Sahar, and T. Waheed, “Real Time Digital Image Processing Using Point Operations in Multithreaded Systems,” ResearchGate, Oct. 2015. https://www.researchgate.net/publication/301443116_Real_Time_Digital_Image_Processing_Using_Point_Operations_in_Multithreaded_Systems.

[14] A. Kika and S. Greca, “Multithreading Image Processing in Single-core and Multi-core CPU using Java,” International Journal of Advanced Computer Science and Applications, vol. 4, no. 9, 2013, doi: https://doi.org/10.14569/ijacsa.2013.040926.

[15] J. Seiler and A. Kaup, “Distributed Parallel Image Signal Extrapolation Framework using Message Passing Interface,” arXiv.org, Jul. 01, 2022. https://arxiv.org/abs/2207.00238.

[16]D. Lin et al., “PARALLELIZATION OF VIDEO PROCESSING: FROM PROGRAMMING MODELS TO APPLICATIONS,” ResearchGate, Dec. 2009. https://www.researchgate.net/publication/224586592_The_parallelization_of_video_processing.

[17] L. Deligiannidis and H. Arabnia, “Parallel Video Processing Techniques for Surveillance Applications | IEEE Conference Publication | IEEE Xplore,” ieeexplore.ieee.org. https://ieeexplore.ieee.org/document/6822105?arnumber=6822105.

[18] A. Merigot and A. Petrosino, “Parallel processing for image and video processing: Issues and challenges,” Parallel Computing, vol. 34, no. 12, pp. 694–699, Dec. 2008, doi: https://doi.org/10.1016/j.parco.2008.09.009.

[19] R. Farias, R. Marroquim, and E. Clua. “Parallel Image Segmentation Using Reduction-Sweeps on Multicore Processors and GPUs | IEEE Conference Publication | IEEE Xplore,” ieeexplore.ieee.org. https://ieeexplore.ieee.org/document/6656179.

[20] S. Saxena, S. Sharma, and N. Sharma, “Image Processing Tasks using Parallel Computing in Multi core Architecture and its Applications in Medical Imaging,” Semantic Scholar, 2013. https://www.semanticscholar.org/paper/Image-Processing-Tasks-using-Parallel-Computing-in-Saxena-Sharma/90bd6dce15c64dcb6623ff71b327caa041ecd293.