Implementation of Sobel filter using CUDA

Akshita Akasapu¹, V Sailaja², G Prasad³

¹²Department of EEE, Amrita School of Engineering, Bengaluru, Amrita Vishwa Vidyapeetham, India.
³Special Projects and Photo processing division, National Remote sensing center, Indian Space research organization, Hyderabad, India.

¹akshitaakasapu@gmail.com, ²v_sailaja@blr.amrita.edu, ³prasad_g@nrsc.gov.in

Abstract. The Sobel filter is an edge detection filter that results in highlighting and focusing the edges. It is used for extracting the image gradient and contours in image processing applications. The data sets we are using in the paper are PGM and TIFF images with different rows and column pixel dimensions. The sobel filter operations used are implemented in C and CUDA programming language and the time taken for the operations in the serial and parallel platforms respectively are analyzed.

Keywords. Image Processing, Edge Detection, Sobel operator, C programming language, CUDA programming language, Parallel Performance, Serial Performance, PGM file, TIFF file.

1. Introduction

Image processing is one of the most powerful tools used in fields with requirements of transmission of data especially in telecommunication, security systems, and remote sensing. The manipulation of the images using various algorithms applied to the input to get the desired output is the aim of image processing [1]. The quality of the image under extreme undesirable and unfavourable conditions can be improved using Digital Image Processing(DIP) in several ways by adjusting the image properties using edge detection, noise reduction, etc [2]. Acquisition, enhancement, restoration, morphological processing, segmentation, representation &description, object recognition are the key stages used in image processing. The main components of DIP are importing, analysis and manipulation and then output. Where scanning and digital photography are used for importing. Analysis and manipulation accomplished using various software applications. Image processing has extensive applications in many areas, including astrophysics, medicine, robotics, and remote sensing by satellites [3][10].

The process of portioning the image into constituent parts or objects using groups of pixels that are homogeneous concerning to some criterion is called Segmentation. The pixel collection in each group must be different and adjacent groups must be heterogeneous. Segmentation algorithms are mainly based and focused on region orientation rather than pixel-oriented. Segmentation finally results in the splitting of images into similar and connected areas. It mainly focuses on the division of images into meaningful regions.

Satellite Sensors collect data in different spectral bands and hence this data is represented in many dimensions. Due to the multiple dimensions, redundancy of data arises. To remove this and to emphasize and isolate the required data from the image, edge detection is applied.

Data is transmitted from satellites to the research centres on the ground with an objective to monitor the Earth's surface and atmosphere. This data is of huge volume and it is very difficult to store, retrieve, and analyze. Edge Detection is the general term for a technique to reduce the dimensionality...
of the image and isolate the required data from the huge data by emphasizing the edges. Using mathematical horizontal and vertical projection, along with gradient operation, the required data is isolated with the required dimensions from the original huge volume data set. It is therefore often the case that the data acquired from the reduced dimension data set will allow the user to spot patterns, obligate and requisite data, way more easily than without performing the Edge detection operation using Sobel Filter Operator.

2. Related work

Image processing is becoming most challenging in the fields like astrophysics, medicine, etc. Different segmentation algorithms can be used for the detection of images [10]. One of the main steps in image processing is the segmentation of the image. The partitioning of the image is called segmentation. The method which helps in separating the foreground and the background of an image for accurate analysis is called image segmentation. Techniques involved in image segmentation are edge base segmentation, Fuzzy based segmentation, Threshold based segmentation and Region based segmentation [9]. Image segmentation can be used in the medical aspect to avoid problems such as noise and signal distribution during processing. But research has been conducted to analyze the algorithm in a wider range of applications as well [11].

Edge detection is a foundation tool in image processing mainly focusing on feature detection and extraction. This mainly aims towards the identification of the pixels in an image where there are sudden disruptions in the pixel values or any sharp differences in the image luminance. Different techniques have been developed and analyzed for the identification of the edges, one of the techniques is to find the edges to the respective image and then identify the image. Depending on the type of edges, several edge detection operators have been developed. Edge orientation, image noise and edge structure are the variables effected in the selection of edge detection operators. The majority of different edge detection methods may be grouped into Gradient and Laplacian [7]. But many times, the algorithms can get overloaded on the machine through series performance and hence, this would result in delayed execution of the algorithm and intensive computational problems when high resolution inputs are used. An effective solution to resolve this problem is by parallel performance.

The graphics processing unit (GPU) provides a constructive solution for the intensive computing of the image during image processing and provides high performance in real time applications. GPU features parallel programming techniques that ensure performance by highest standards and reduce the execution time of accelerating an extensive calculation algorithm [6].

3. Edge detection technique

The sobel operator is a derivative mask and is used for edge detection. Sobel operator is used to detect the horizontal and vertical edges in an image. Figure 1 explains the working of the filter.
3.1. Horizontal and Vertical Gradient Components

![Flowchart of Sobel Filter](https://example.com/flowchart.png)

**Figure 1** Flowchart of Sobel Filter

The horizontal gradient component ($G_x$) in figure 2 and vertical gradient component ($G_y$) in figure 3 are used to find the edges oriented to the respective operations. The operators can be applied separately to find the respective direction of edges or can be made into a single direction by finding the absolute magnitude gradient. The absolute magnitude is given by equation 1[7].

$$ |G| = \sqrt{G_x^2 + G_y^2} $$

(1)[7]

The angle of orientation of the edge is formulated by equation (2)[7].

$$ \theta = \arctan(G_y/G_x) $$

(2)[7]

3.2. Pseudo Convolution Operator

Two gradient components are computed and added in a single pass to obtain the absolute magnitude using the pseudo-convolution operator shown in Figure 3.
Quick computation of approximate gradient magnitude is obtained from Pseudo-convolution kernel as shown in Figure 4. The approximate magnitude is given by equation (3)[8].

\[ |G| = |(A_1 + 2 \times A_4 + A_7) - (A_3 + 2 \times A_6 + A_9)| + |(A_2 + 2 \times A_8 + A_9) - (A_1 + 2 \times A_2 + A_3)| \]

4. Implementation details

Figure 5 gives a brief description of how the implementation is carried forward and the various software and file formats used.

4.1. Block 1 - Gradient Edge Detection Filter

In order to increase the clarity of an image, we have to find all the edges. Edges can be identified by applying algorithms called edge detection filters. Once all the edges have been identified, they can be applied to the image for making it sharper. In this paper, we have implemented this Edge Detection Filter on a serial platform on a Central Processing Unit (CPU) using C Programming Language and Parallel Platform on a GPU using the CUDA programming language.

4.2. Block 2 - Platforms Used

4.2.1. Serial Processing: It is a method where a storage device moves through all the locations from where it aims to read or write and gathers information and then the data is retrieved from the device.
4.2.2. **Parallel Processing**: Process of evenly assigning computer processes between computer processors or cores. This requires a computer with multiple CPUs or a single CPU or GPU equipped with multiple cores. An operating system capable of supporting parallel processing, or software is written specifically to process instructions in parallel are essential.

4.3. **Block 3 - Test Implementation**
The Sobel operator is applied to PGN files chosen as test datasets.

4.4. **Block 4 - Implementation on Satellite image**
The data from the satellite arrives in the form of TIFF images. The obtained TIFF image is executed using the sobel edge detection operator.

5. **Results**
5.1. **Serial processing using static approach**
The C program has been tested with multiple inputs. Sobel operator is applied to the figure 6, using Serial Processing C code and the time taken for the execution from figure 8 is noted. The figure 7 shows the output image after application of the filter.

![](image1)

**Figure 6** Input of serial processing static approach  
**Figure 7** Output after edge detection

Time taken to execute the filter operation is: 0.115749 seconds

5.2. **Serial processing multi thread approach**
The C program has been tested with multiple inputs. Sobel operator is applied to the figure 9 using Serial Processing C code and the time taken for the execution from figure 11 is noted. The figure 10 shows the output image after application of the filter.
5.3. Parallel processing
The CUDA program has been tested with multiple TIFF inputs. Sobel Operator is applied to the figure 12 and figure 15 using the parallel processing CUDA code and figure 13 and figure 16 is the output image obtained. Then the time taken for execution from figure 14 and figure 17 is noted.

![Figure 9](image1.png) Input of serial processing multi thread approach

![Figure 10](image2.png) Output after edge detection for 5 threads

![Figure 11](image3.png) Time taken for 5 thread execution

![Figure 12](image4.png) Input of parallel processing

![Figure 13](image5.png) Output after edge detection

![Figure 14](image6.png) Time taken for execution
6. Observations

The paper is focused on CUDA and its parallel programming platform and the time gained is reviewed. The time gained is calculated using an edge detection filter algorithm. A well-known algorithm SOBEL for edge detection has been used as a part of the paper. A dataset of images is tested using both serial and parallel processing algorithms and the results are noted and observed. The results are further divided based on the image pixels used and the device used.

| Table 1 Serial processing using static approach |
|-----------------------------------------------|
| Input image name | Size (pixels) | Time (Seconds) |
|------------------|---------------|----------------|
| 1 Lena.pgm       | 1024X1024     | 0.119251       |
| 2 Man.pgm        | 1024X1024     | 0.115749       |
| 3 House.pgm      | 1024X1024     | 0.114897       |

From table 1, the results acquired after implementing the sobel edge detection filter, the following are the conclusions:

- On comparison with the multithreading approach with the constant pixel size input image, the static approach takes 0.014398 seconds more time for the operation.

- As the complexity of the input image increases, the number of edges increase and hence the time taken to find edges using technique decreases.

- It is observed that the time taken (in seconds) are in the order,
  
  LENA.PGM > MAN.PGM > HOUSE.PGM

  Since the number of edges are in the order,

  LENA.PGM < MAN.PGM < HOUSE.PGM

- Hence, for images with more edges, it is comparatively easier to detect the edges and emphasize the image and hence the time taken is less.
Table 2 Serial Processing Using Multi threading Approach

| Input image name | Size (pixels) | Threads | Time (Seconds) |
|------------------|--------------|---------|----------------|
| Lena.pgm         | 512x512      | 1       | 0.028458       |
|                  |              | 2       | 0.053174       |
|                  |              | 3       | 0.043227       |
|                  |              | 4       | 0.1025         |
|                  |              | 5       | 0.050856       |

Table 3 Serial Processing Using Multi threading Approach

| Input image name | Size (pixels) | Threads | Time (Seconds) |
|------------------|--------------|---------|----------------|
| Man.pgm          | 1024x1024    | 1       | 0.101351       |
|                  |              | 2       | 0.183654       |
|                  |              | 3       | 0.159045       |
|                  |              | 4       | 0.219699       |
|                  |              | 5       | 0.212515       |

From table 2 and table 3, the results acquired after implementing the sobel edge detection filter, the following are the conclusions:

- On comparing the two input images of different pixel sizes, we can observe that, for smaller pixel images, the time difference for multiple threads is constant (i.e. almost negligible).

- For the image with a larger pixel size, it is observed that as the number of threads increase, more strips are to be computed. Hence, the time taken also increases uniformly.

- Hence, as the image resolution and pixels increase, the time taken for the operation also increases uniformly along with the increase in the number of threads.

Table 4 Parallel Processing

| Input image name | Size (pixels) | Time (Seconds) |
|------------------|--------------|----------------|
| 1 Lena.tiff      | 512X512      | 0.334          |
| 2 City.tiff      | 577X419      | 0.568          |
| 3 River.tiff     | 256X256      | 0.203          |
| 4 Satellite.tiff | 571X579      | 0.601          |

From table 4, the results acquired after implementing the sobel edge detection filter, the following are the conclusions:

- From the obtained data, it is observed that, for an 8 X 8, i.e. 64 threads CUDA code, as the pixel’s increase, as shown below,

  RIVER.TIFF < LENA.TIFF < CITY.TIFF < SATELLITE.TIFF
The time taken (in seconds) also increases.

- It is observed that the time taken to compute the filter operation on the same Lena data set of 512X512 pixels on a GPU is greater than on CPU.

7. Conclusion

- CPU has a lesser number but more powerful cores compared to GPU.

- For smaller pixel images, the CPU gives faster outputs compared to the GPU, but the GPU gives better results.

- For larger scale and high resolution data sets, GPU is preferred due to the capability of powerful parallel instruction processing techniques.

8. References

[1] Jain A Namdev and M Chawla 2016 Parallel edge detection by SOBEL algorithm using CUDA C (Bhopal)

[2] R C Gonzalez and R E Woods 2009 Digital Image Processing (New Delhi, India).

[3] https://www.britannica.com/technology/image-processing

[4] Vincent O R and Folorunso O 2009 A descriptive algorithm for sobel image edge detection (California: Informing Science Institute ) Vol. 40 pp. 97-107.

[5] Gupta, Samta, Susmita Ghosh Mazumdar and M Tech Student 2013 Sobel Edge Detection Algorithm.

[6] H B Fredj, M Ltaif, A Ammar and C Souani 2017 Parallel implementation of Sobel filter using CUDA (Hammamet) 2017, pp. 209-212.

[7] Vairalkar, Manoj K, and S U Nimhorkar 2012 Edge detection of images using Sobel operator.

[8] https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm

[9] J N Chandra, B S Supraja and V Bhavana 2017 A Survey on Advanced Segmentation Techniques in Image Processing Applications (Coimbatore) pp. 1-5.

[10] J N Chandra, V Bhavana and H K Krishnappa 2018 Brain Tumor Detection Using Threshold and Watershed Segmentation Techniques with Isotropic and Anisotropic Filters (Chennai) pp. 0372-0377.

[11] P Tyagi, T Singh, R Nayar and S Kumar 2018 Performance comparison and analysis of medical image segmentation techniques (Bangalore) pp. 1-6.

[12] Sanders, Jason, and Edward Kandrot 2011 CUDA by Example: An Introduction to General-Purpose GPU Programming (Upper Saddle River, NJ Addison-Wesley).

[13] T Naqash and I Shafi 2011 Edge sharpening in grayscale images using modified Sobel technique (Karachi) pp. 132-136.