De La Salle University, Manila
Computer Technology Department

Term 2, A.Y. 2024-2025

CEPARCO -

Project Proposal:
Implementation of Canny-Edge Detection using CUDA 12 Features

GROUP 4

Arca, Althea Denisse G.

February 24, 2025

Project Proposal: Implementation of Canny-Edge Detection using CUDA 12 Features

Abstract: The aim of this proposal is to use CUDA's parallel computing capabilities to increase the Canny edge detection algorithm's computational efficiency. In order to improve important computing phases like Gaussian filtering, gradient calculation, non-maximum suppression, and hysteresis thresholding, the technique will be implemented in both sequential C and parallelized CUDA versions. By contrasting execution times and analyzing the precision of the edges that are detected, the performance of the two implementations will be evaluated. Demonstrating that GPU acceleration is a feasible method for real-time image processing applications by significantly reducing execution time while keeping or improving edge detection quality is the aim.

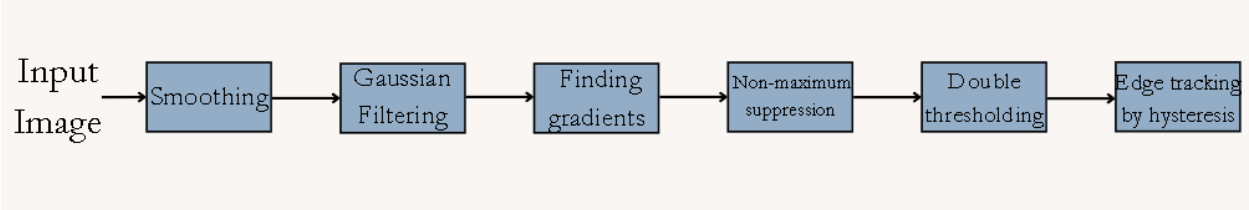
Project Description

Edges are a feature that are found in an image as they provide context and objects of images. They are used in image segmentation and data extraction; important in the field of image processing, computer vision, and machine vision (Babasaheb Bhimrao Ambedkar University, n.d.). Hence, the needs of edge detection algorithms. Examples of these edge detections algorithms are Roberts operator, Sobel operator, and LoG operator. These examples that are listed are simple, but their detection accuracy is poor and provide little denoising (Liu et al., 2022).

In 1986, A new algorithm developed by Canny that can improve the accuracy and quality of edge detection by providing a multistep algorithm that better edge detection, and helps with denoising (Canny, 1986). In this proposal, the team will be comparing the execution and processing time of this algorithm while using the Nvidia's CUDA cores. Afterwards, analysis is done to ensure its quality and to see a significant improvement in performance speed.

Proposed Algorithm

The computationally intensive phases of Gaussian filtering, gradient computation, non-maximum suppression, and hysteresis thresholding are all part of the Canny edge detection technique. In order to take advantage of the parallel processing capability of GPUs, we will optimize and parallelize these procedures using CUDA in this project.



First, Gaussian filtering is used to filter the input image and remove noise through a convolution operation with a Gaussian kernel. Second, during gradient computation, we both calculate the magnitude and direction of the gradient using Sobel operators. The non-maximum suppression stage is next used to thin

out the edges by keeping only the highest gradient responses. This stage involves accessing neighboring pixels along the gradient direction, which can lead to memory access inefficiencies. Lastly, hysteresis thresholding is applied to identify edge pixels as strong or weak depending on connectivity. Historically, this is done using recursive operations or region-growing methods that are sequential in nature.

The project will illustrate how CUDA's Unified Memory, prefetching, and memory advice capabilities can be employed to enhance the performance of the Canny edge detection process. The end results will contrast execution times for sequential versus parallel versions, revealing the kinds of speedups and efficiency enhancements enabled by contemporary CUDA features in live image processing applications.

Proposed target implementation platform of the proposed project

The target implementation platform of this proposed project will use a nVidia GPU with CUDA. and the development environment will be set up using CUDA 12 toolkit and Visual Studio on a windows operating system. this should provide a solid support for compiling and debugging CUA-enabled C++ code.The primary programming language will be C/C++ with cuda extensions, allowing efficient parallelization and direct access to GPU resources for high-performance computing tasks.

Mid-project milestones of the proposed project

The milestones for the projects, there will be 4 big milestones ahead for it. First, the recreation of Canny Edge detection algorithms in C and implementation within CUDA cores. Another is the Initial benchmarking and comparison to compare and contrast the execution time and performance. Third, code clean up and optimization. This part is where the team cleans up the code and better uses the GPU performance to ensure more utilization. Lastly, documentation and report on finding, to understand and achieve performance from the CUDA.

References

[1] Ashok, V. (2010, July 13). *Canny Edge Detection in C#*. Codeproject.com; CodeProject. https://www.codeproject.com/Articles/93642/canny_edge_detection/canny_edge_detection_c_.zip

[2] Mao, S., Liu, Y., & Xiao, L. (2022). Improved Canny edge detection algorithm for image segmentation. In Proceedings of the 2021 International Conference on Sensor Design, Manufacturing and Computational Intelligence (SDMC-21). Atlantis Press. https://www.atlantis-press.com/article/25873669.pdf

[3] Canny J. (1986). A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis & Machine Intelligence, 8(6):679-98.

[4] Babasaheb Bhimrao Ambedkar University. (n.d.). *Edge detection*. Retrieved from https://www.bbau.ac.in/dept/CS/TM/Edge%20detection%20.pdf