**Performance optimization for high resolution image processing by spatial domain filtering on Android OS based mobile systems using multi-threading.**

***Abstract***

**Introduced an application for Android based mobile smart devices which demonstrates a method to process high resolution images in spatial domain and increasing performance efficiency using multithreading.**

**Laplacian enhancement (3x3) and uniform weighted smoothing (3x3) filter kernels are used to study and benchmark comparison of single main thread processing and multi thread processing.**

**Benchmarking is done based on time required for RGBA data extraction and to process image by applying the above-mentioned filter kernels respectively.**

**Three test cases with varying pixel count of images are demonstrated to replicate the performance improvement results.**

***Keywords***

**Multi-threading, Android activity**

1. Introduction

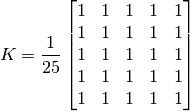
Image processing by spatial domain filtering is useful in enhancing and highlighting the important features as well as smoothing out over-exposed and noisy images depending on type of filter kernel. Image processing is a vast area whose applications are in many fields in which autonomous cars, autonomous drones, optical character recognition and medical imaging are the trending fields.

In 2018, mobile smartphones are powerful than ever surpassing the computational power of an average dual core computer.

This paper is providing a method to use that mobile computational power to process high resolution images on the go with better performance efficiency to develop the applications of portable and real-time image processing using multi-threading to perform large number of arithmetic operations required for application of spatial domain filters parallelly in synchronization on multiple threads.

All the methods described in this paper are based on the constraints and security of Android operating system.

Uniform Weighted (5x5) smoothing filter is used to remove noise and sharp variations of the RGB values of every pixel compared to their neighbouring pixels respectively. A 5x5 matrix (Figure: 1.1) is used to average the RGBA values of surrounding pixels by element wise multiplication for each pixel of the image.



*Figure: 1*

The Laplacian is a 2-D isotropic measure of the 2nd spatial derivative of an image. The Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection

(example: Figure 1.2 b)

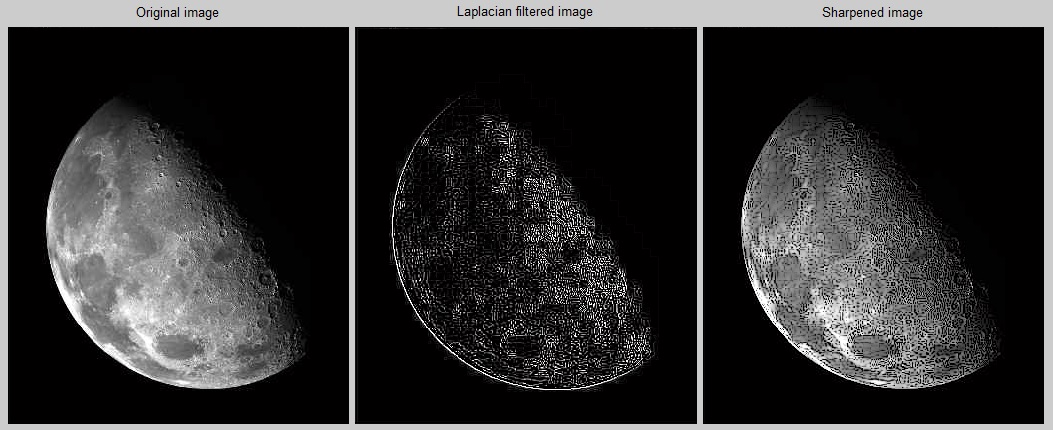


*(Laplacian function)*

*x: x-coordinate of pixel*

*y: y-coordinate of pixel*

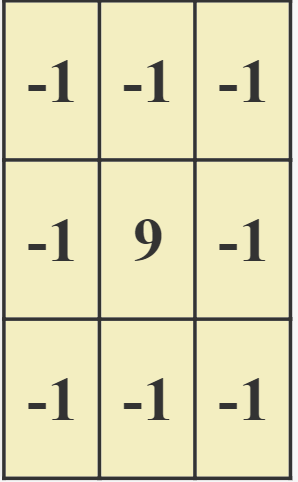
Laplacian enhancement is obtained by removing the RGBA values obtained from Laplacian filtering from original image. This process results in an enhanced image (Figure: 1.2 c)



*a. original b. Laplacian filtered c. Laplacian enhanced*

*Figure: 1.2*

The resultant variation of filter kernel to obtain Laplacian enhanced image is shown in Figure:1.3.



*Figure: 1.3*

1. Methodology

There are 2 components required for image processing on Android:

1. Bitmap Data extraction
2. Arithmetic multiplications on the extracted data for applying filter

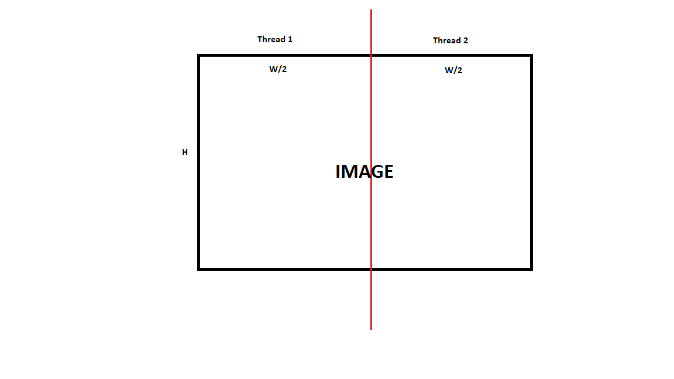
In both processes, multi-threading can be used to reduce time consumption.

The idea is to divide the given high-resolution images into smaller chunks of image to process every chunk simultaneously by assigning each chunk a thread and running these threads parallelly.

The number of pieces in which the image is divided is decided according to the hardware specification and use case of it.

Data extraction from an image involves extraction of Red, Green, Blue and Alpha (RGBA) values of each pixel and store then in their respective arrays.

For demonstration purposes, 2 threads are used for data extraction and each thread extracts data from their respective half of the image. (Figure: 2.1)



*Figure: 2.1 Thread 1 and 2 extract data from their respective half simultaneously. Image Resolution: W X H*

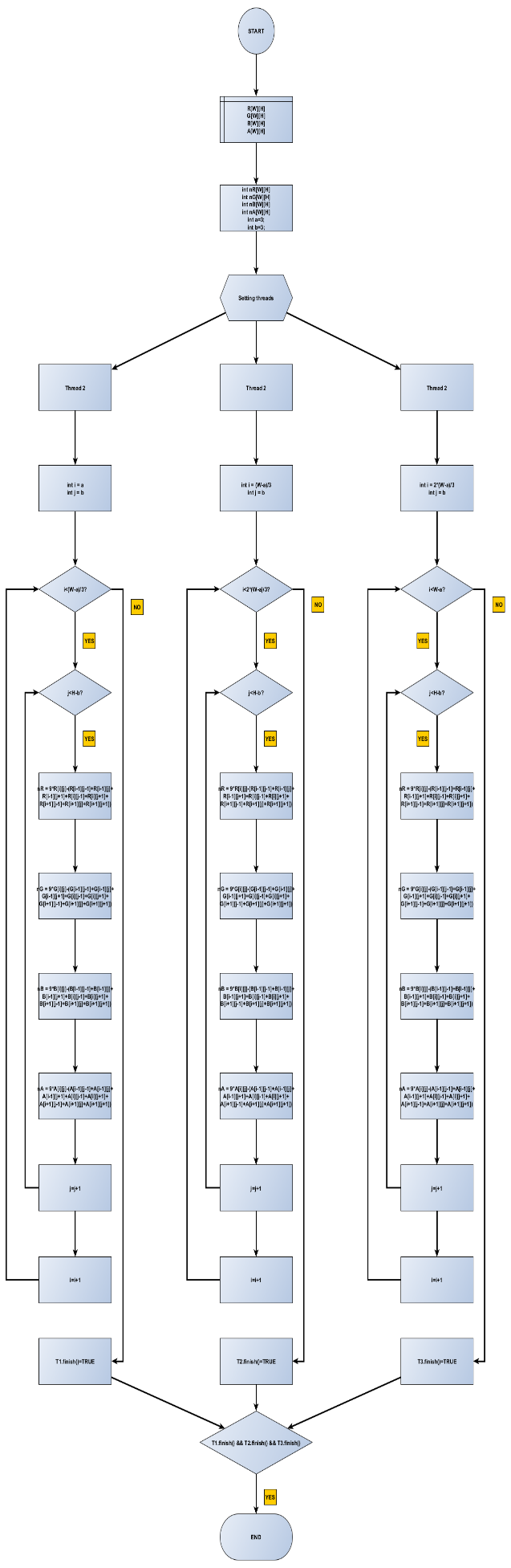
*Flowchart for image data extraction using multi-threading (2 threads):*

*A screenshot of a cell phone

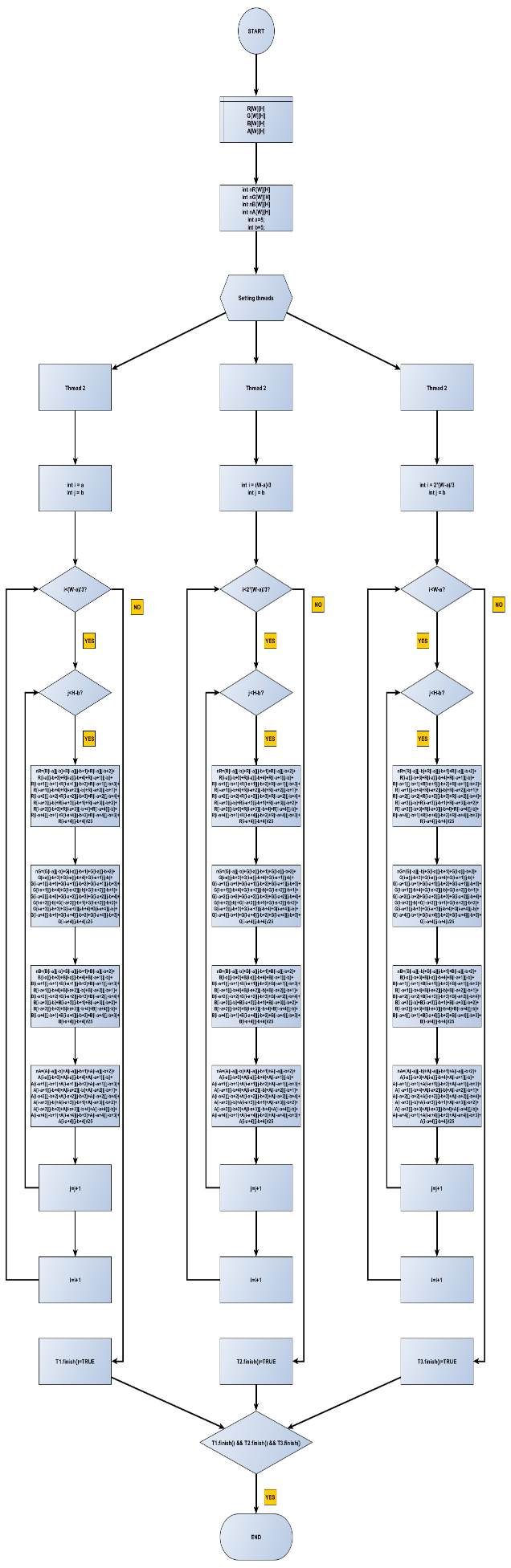
Description generated with high confidence*

For demonstration purposes, 3 threads are used for arithmetic multiplication for applying spatial domain filters:

*Flowchart for application of (3x3) Laplacian enhancement filter on extracted data using multi-threading (3 threads):*

**

*Flowchart for application of (5x5) Uniform Weighted Smoothing filter on extracted data using multi-threading (3 threads):*

**

1. Results

**Testing Hardware Specifications:**

**OS**: Android 8.0 (Oreo)

**Chipset**: Exynos 8895 Octa (10 nm)

**CPU**: Octa-core (4x2.3 GHz Mongoose M2 & 4x1.7 GHz Cortex-A53)

**GPU**: Mali-G71 MP20

TEST IMAGES:

Image 1: (Figure: 3.1)



A picture containing screenshot

Description generated with high confidence

*Figure: 3.1 Performance comparison graph (Image 1)*

Image 2: (Figure: 3.2)

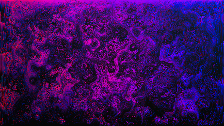


A screenshot of a video game

Description generated with high confidence

*Figure: 3.2 Performance comparison graph (Image 2)*

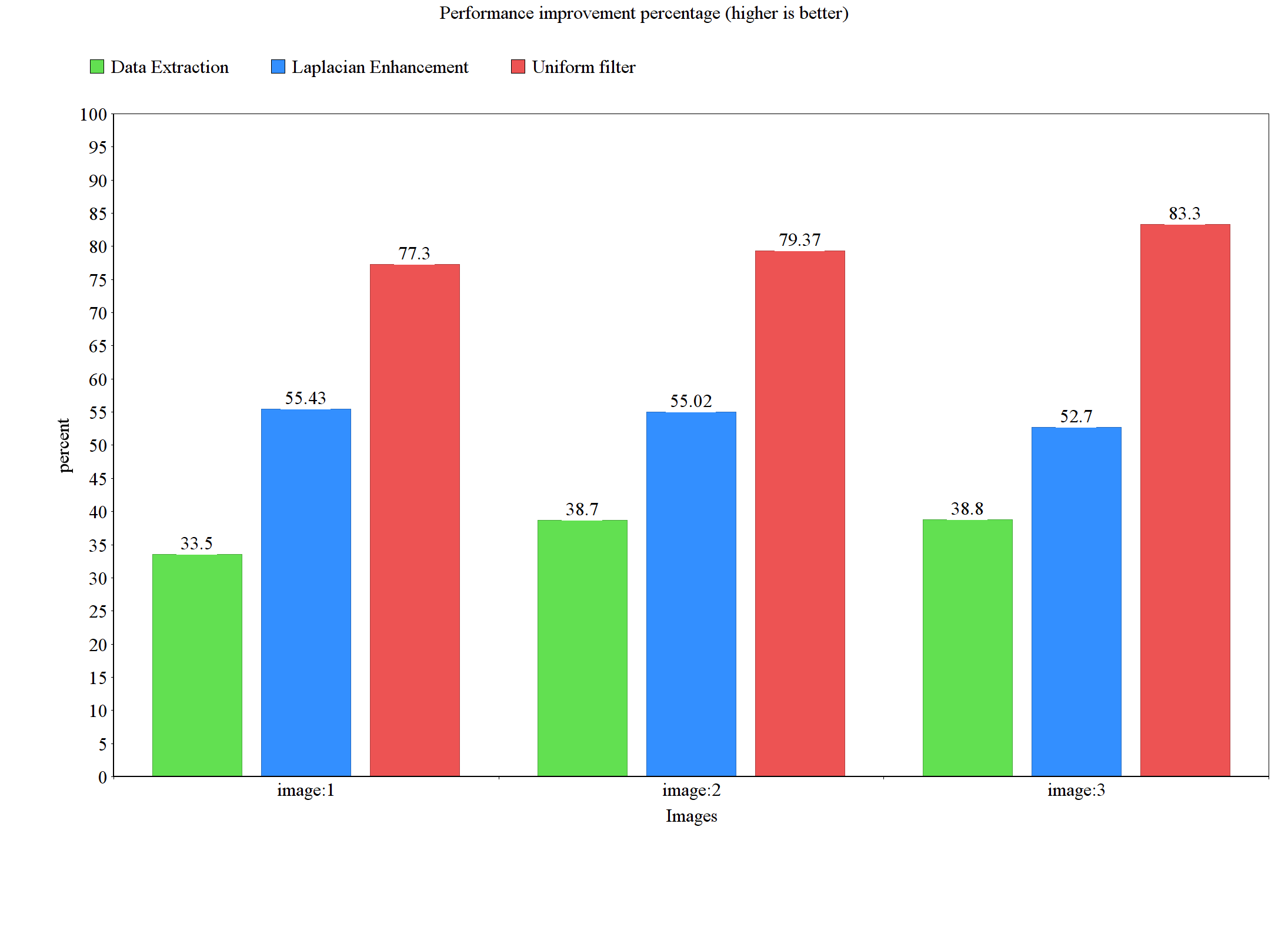
Image 3: (Figure: 3.3)



A screenshot of a computer

Description generated with very high confidence

*Figure: 3.3 Performance comparison graph (Image 3)*

*Figure: 3.4 Percentage Improvement comparison of test images.*