Object- Oriented Programming

Specification Project

Abdul Razaak Abdul nabi A.

Digital Image Processing

1. project Description:

Digital image processing is the use of a digital computer to process digital images through an algorithm. As a subcategory or field of digital signal processing, digital image processing has many advantages over analogue image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be modelled in the form of multidimensional systems. The generation and development of digital image processing are mainly affected by three factors: first, the development of computers; second, the development of mathematics (especially the creation and improvement of discrete mathematics theory); third, the demand for a wide range of applications in environment, agriculture, military, industry and medical science has increased.
In the end, this program capable to do few things like use 2D image and change the “Brightness: changing the intensity of the image by manipulating the main 3 colours (RGB)”, “Binarization: Converting a pixel image to a binary image”, “Some types of Filters, like Range and Median, ect….” and “Contrast: Difference in luminance or colour that makes an object distinguishable”, etc…

2. Mathematical model:

1. Histogram:

$$n = \sum_{i=1}^{k} m_i$$

n = Total number of Observations, mi = Histogram Function, k = Total number of bins.

$$M_i = \sum_{j=1}^{i} m_j$$

Mi = Cumulative Histogram.

Histogram is discrete function $p(f) = \frac{n_f}{n}$,
n_f represents number of pixels with intensity f
n is a number of all pixels,

Histogram $p(f)$ defines probability of each intensity appearance in image.

2. ***Contrast***:

$$C = \frac{I_{diff}}{I_{Ave}}$$

C = Contrast, I_{diff} = Luminance difference, I_{Ave} = Luminance Average.

$$g(i,j) = a * f(i,j)$$

a : defines discrimination of details in image with different intensity.
intensity of each pixel is multiplied by value a
$f(i,j)$ input image

$g(i,j)$ output image

If a>1 then higher contrast
If a<1 then lower contrast
(b) = intensity

$$g(i,j) = a * f(i,j) + b$$

To calculate the contrast ratio, the relative luminance of the lighter colour (L1) is divided through the relative luminance of the darker colour (L2):

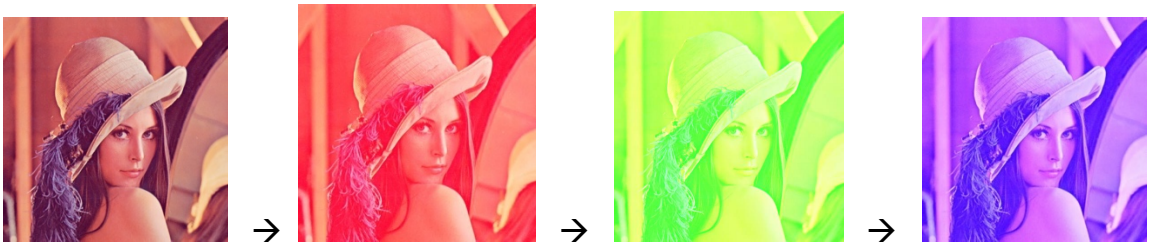
$$(L1 + 0.05) / (L2 + 0.05)$$

This results in a value ranging from 1:1 (no contrast at all) to 21:1 (the highest possible contrast). While high contrast is generally good, higher is not always better.

3. ***Brightness:***

$$\mu = \frac{R + G + B}{3}$$

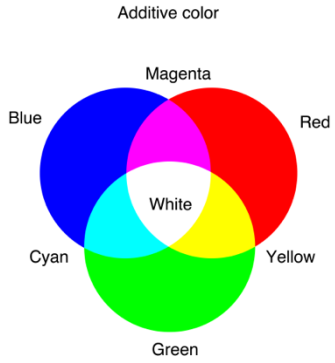
μ =Arithmetic mean of R, G, B color coordinates.
From main colours (Red, Green, and Blue) one can create derivative colours by addition.
To start mixing in RGB, think of each channel as a bucket of red, green, or blue paint. With 8 bits per channel, you have 256 levels of granularity for how much of that colour you want to mix in; 255 is the whole bucket, 192 = three quarters, 128 = half bucket, 64 = quarter bucket, and so on. Once in the ballpark, you can adjust the mix by finer increments.



Original Image → Red component → Green component → Blue component

preserving the channel ratios across colours and finding ways to recombine and riff on them can create more natural colour harmonies amongst all of the swatches that end up in my palette. There's something poetic about building colour palettes by using the parameters of the colour space itself as mixing model.

What *additive* means is that adding more colour information from any channel adds light to make brighter colours, ultimately resulting in white. You can see the summation of colours resulting in white in the colour wheel graphic above as well as the diagram below.



The analogous nature of the colours is enforced by our logical use of the additive colour chart. Essentially, the result of inverting the channel values was to reflect our original colour across the cyan axis of the chart.

4. ***HSL or HSV (color):***

$$M = \max(R, G, B)$$
$$m = \min(R, G, B)$$
$$C = range(R, G, B) = M - m$$

M = Maximum, m = minimum, C = Chroma.

Hue, Saturation, and Lightness (**HSL**) is a colour model that is often used in place of the RGB colour model in graphics and paint programs. In using this colour model, a colour is specified then white or black is added to easily make colour adjustments.
HSL may also be called HSB (short for hue, saturation and brightness).

More precisely, both hue and chroma in this model are defined with respect to the hexagonal shape of the projection. The *chroma* is the proportion of the distance from the origin to the edge of the hexagon. This ratio is the difference between the largest and smallest values among *R*, *G*, or *B* in a colour. To make our definitions easier to write, we'll define these maximum, minimum, and chroma component values as *M*, *m*, and *C*, respectively.

The *hue* is the proportion of the distance around the edge of the hexagon which passes through the projected point, originally measured on the range [0, 1] but now typically measured in degree [0°, 360°]. For points which project onto the origin in the chromaticity plane (i.e., greys), the hue is undefined. Mathematically, this definition of hue is written:

$$H' = \{undefined, if\ C = 0\ ,\ \begin{cases} \frac{G-B}{C}\ mod\ 6, & if\ M = R \\ \frac{B-R}{C} + 2, & if\ M = G \\ \frac{R-G}{C} + 4, & if\ M = B \end{cases}$$

$$H = 60^{\circ} \times H'$$

Sometimes, neutral colours (i.e. with *C* = 0) are assigned a hue of 0° for convenience of representation.

In the HSV "hex cone" model, *value* is defined as the largest component of a colour. Forming a Hexagonal pyramid out of the RGB cube.

$$V = \max(R, G, B) = M$$

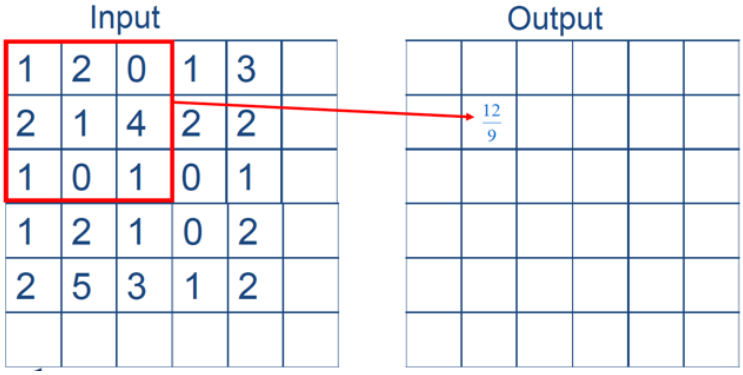
So, the final form written:

$$H = \begin{cases} 0 & if\ Max = Min \\ 60^{\circ} \times \dfrac{G-B}{Max-Min} + 0^{\circ} & if\ Max\ R = nd\ G \geq B \\ 60^{\circ} \times \dfrac{G-B}{Max-Min} + 360^{\circ} & if\ Max\ R = nd\ G < B \\ 60^{\circ} \times \dfrac{B-R}{Max-Min} + 120^{\circ} & if\ Max = G \\ 60^{\circ} \times \dfrac{R-G}{Max-Min} + 240^{\circ} & if\ Max = B \end{cases}$$

$$S = \begin{cases} 0 & if\ Max = Min \\ \dfrac{Max-Min}{Max+Min} = \dfrac{Max-Min}{2l} & if\ l \leq \dfrac{1}{2} \\ \dfrac{Max-Min}{2-(Max+Min)} = \dfrac{Max-Min}{2-2l} & if\ l > \dfrac{1}{2} \end{cases}$$

$$L = \frac{1}{2}(Max + Min)$$

5. **Filters:**



A. Median

The median "m" of a set of values (e.g. image pixels in the filtering mask) is such that half the elements in the set are less than "m" and other half are grater than "m".

$$x(n) = \{1, 5, -7, 101, -25, 3, 0, 1, 7\}$$

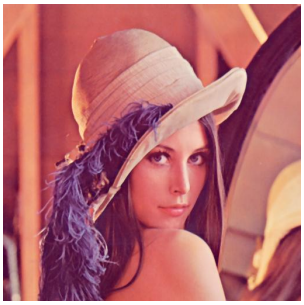
Sorted sequence f elements:

$$x_s(n) = \{-25, -7, 0, 1, 3, 5, 7, 11, 101\}, 3\ is\ median$$

Median filtering the image:

$$Image\ source\ f \rightarrow \left[\overset{h}{\quad f(x,y) \quad} \right] \rightarrow (g(x,y) = median\{f(x,y); (x,y) \in h\}) \rightarrow \left[\quad g(x,y) \quad \right] \leftarrow output\ Image$$

The most time consuming and operation s sorting, values of the output image are equal or smaller than the values of the input image (no rescaling)…



B. Range:

Range filtering refers to filtering using the values of the pixels (rather than their positions). More precisely, the weights of the filter, in this case, would decay from the centre based on "distance" between the values of the pixels. There are also filters that perform a combination of these.
The local range is calculated by ordering the N pixels within the window according to intensity, i.e.:

$$(f1, f2, f3, \dots\dots\dots, fN)$$

Where:

$$f1 \leq f2 \leq \dots\dots.. \leq fN$$

and then subtracting the intensity values for two selected positions ($i\ and\ j$) within this ordered list such that:

$$range\ (j, i) = fj - fi \qquad 1 \leq i < j \leq N$$

When performed over the whole image, this may be represented by

$$g = Raj, i(f)$$

where g is the output image. Thus, the range filter IS an extension of another local nonlinear filter, the rank. filter, defined as

$$rank(i) = fi \qquad 1 \leq i \leq N$$

When performed over the whole image, this may be represented by:

$$g = Ri(f)$$

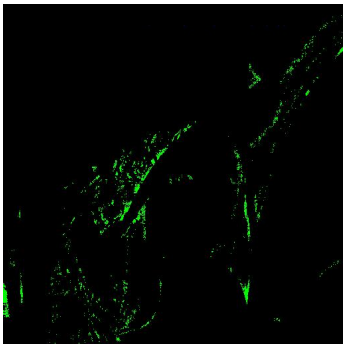
Therefore:

$$Raj, i(f) = Rj(f) - Ri(f) \qquad i < j$$

Thus, a range-filtered image is the difference of two. rank-filtered images. Some of the properties of the range filter may be derived from the properties of the rank filter.

6. **Binarization:**

Binarization is the process of converting a pixel image to a binary image. Image binarization is the process of separation of pixel values into 2 groups, white as background pixel and black as foreground or object pixel, or separated it into 3 groups Red, Green and Blue… (Using the calculation of the Brightness and manipulated the equations). By using the RGB threshold (Adaptive Threshold), it first counts the appearance of each tone in the image and tries to find a good centre.



3. Graphical interface User:

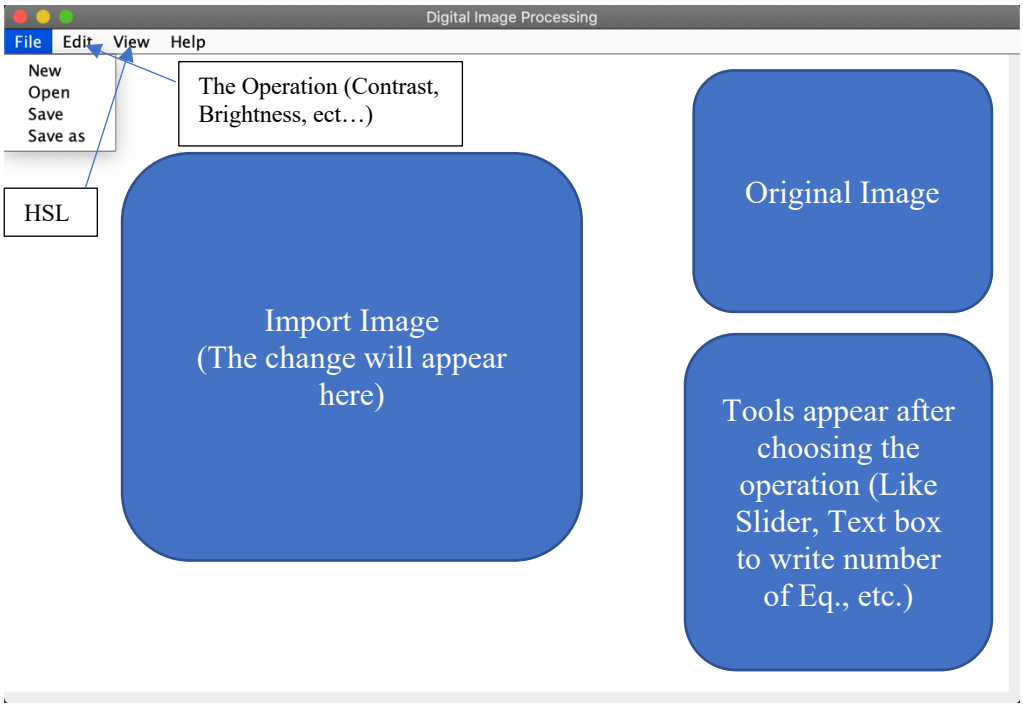


Fig. 1 Sketch of the user interface of the main program window.

4. Usage scenarios:

1. Run the Software.
2. Loading an image (Menu/File → Open → [Choose the prefer image] → Add).
The image will appear in panel.
3. The simulation:
Contrast, Filters, Binarization, Histogram and Brigthness.
(Edit → Contrast
→ Filters → Median
→ Average
→ Range
→ Binarization
→ Histogram
→Brigthness)
Change the Intensity of the image color:
(View → HSL)
4. Save the simulation performed:
Menu/File → Save as
5.Clear all data or simulation
Edit → Reset.
6. The program is closed by clicking on the cross in the upper right corner or by pressing:

Menu/File → Close or by pressing the botton (Close).

5. Additional requirements

A. Using the GitHub version control in the project.
B. The program is multilingual (Arabic and English language versions/*Maybe/Possible*).

6. Project Schedule:

I. Specification (3 classes)
II. Prototype - User Interface (5 classes) - ready user interface
III. Release Candidate (10 classes) - implementation of a minimum of half the functionality
IV. Final (15 classes) - a completed project that meets the assumed requirements.

7. Assessment & Task Table:

Task table template:

Project task table "Digital Image Processing"

Number	**Functionality**	**Maximum number of points**	**The number of points obtained**	**Notes**
1	GUI			Maybe will be logo when it's open.
2	The program on GitHub.	*Mandatory*		Started using it on 30.03.2020
3	Contrast			
4	*Histogram*			*Mandatory*
5	Brightness			
6	*HSL*			
7	Binarization			
8	Filter Median			
9	Filter Range			
10	Reset			

	Amount of points:			

Za poprawnie wykonany projekt chcielibyśmy uzyskać ocenę (5) .

Number	**Functionality**	**Maximum number of points**	**The number of points obtained**	**Notes**
1	Add Image			
2	Subtract Image			
3	Face detection			
4	Edge Detection			
5	Different types of Filters			
6	Paint			
7	Other ideas: calculate the Focal length and spot, Aberration, ect….			
8	Multilingual program			*Maybe/Possible*
9	Amount of points:			

These functionalities above in the 2nd Table Maybe will be done or not depend on the time (but most accurate will be in the future). No to mention, this specification will be updated on the future, 2 weeks before the submit the final project…