There is tremendous growth in the volume of images on the internet and desktops due to digitization of data and advances in technology. Also, there exists demand for enhancement, conversion of image formats, security, noise removal, smoothening, etc. for the digital image processing. User in today scenario uses various tools for different image processing features while operating on a same digital image. In this paper we discuss, present available tools and then propose a new user friendly, simple, quick, interactive, portable and efficient image processing tool Photo vista. Experimental results show that user can perform different image processing operations using this single integrated tool.

Keywords: Edge Detection, Filters, Image Encryption & Decryption, Math Functions, Smoothening.

1. INTRODUCTION

The aim of image enhancement is to improve the interpretability or perception of information in images for human viewers, or to provide ‘better’ input for other automated image processing techniques. Unfortunately, there is no general theory for determining what ‘good’ image enhancement is, when it comes to human perception. If it looks good, it is good! However, when image enhancement techniques are used as pre-processing tools for other image processing techniques, then quantitative measures can determine which techniques are most appropriate.[4][8] Image enhancement is the improvement of digital image quality (wanted e.g. for visual inspection or for machine analysis), without knowledge about the source of degradation. If the source of degradation is known, one calls the process image restoration.

Both are identical processes, viz. Both input and output are images.[2][6] Many different, often elementary and heuristic methods are used to improve images in some sense. The problem is, of course, not well defined, as there is no objective measure for image quality.[10] Here, we discuss a few recipes that have shown to be useful both for the human observer and/or for machine recognition. These methods are very problem-oriented: a method that works fine in one case may be completely inadequate for another problem.

2. RELATED WORK

Research work has been conducted over few related Image Processing Tools which have been mentioned in the following content:

2.1 Free Image 3.11

Free Image is easy to use, fast, multithreading, safe image enhancing tool compatible with all 32-bit versions of Windows, and cross-platform. Free Image supports popular graphics image formats like PNG, BMP, JPEG, TIFF and others as needed by today's multimedia applications. It supports Loading and saving of as many bitmap types as possible; Easy access to bitmap components, such as palettes and data bit; Converting bitmap’s bit depths from one to another; Accessing pages in a bitmap when there are multiple, such as in TIFF; Basic manipulation of bitmaps, such as rotation, flipping and re-sampling or point operations such as brightness and contrast adjustment; Alpha compositing and alpha blending. However it doesn’t support advanced image processing operations such as convolution and transforms, Bitmap Drawing & Vector Graphics.

2.2 Image Magic

Image Magic is a software suite to create, edit, and compose bitmap images. It can read, convert and write images in a variety of formats including. Use Image Magic to translate, flip, mirror, rotate, scale, shear and transform images, adjust image colors, apply various special effects, or draw text, lines, polygons, ellipses and Bezier curves. Image Magic consists of more than 350,000 lines of C code and optionally depends on several million lines of code in dependent libraries (e.g. JPEG, PNG, TIFF libraries). Given that, one might expect a huge architecture document. However, a great majority of image processing is simply accessing pixels and its metadata and our simple and elegant implementation makes this easy for the Image Magic developer. We discuss the implementation of the pixel cache and getting and setting image properties and profiles in the next few sections. Next, we discuss using Image Magic within a thread of execution. In the final sections, we discuss image coders to read or write a particular image format followed by a few words on creating a filter to access or update pixels based on your custom requirements.

<table>
<thead>
<tr>
<th>Features</th>
<th>Photo vista</th>
<th>Free Image</th>
<th>Image Magic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portable</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Executable File</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Language</td>
<td>Advanced Java</td>
<td>C</td>
<td>C++</td>
</tr>
<tr>
<td>MATH Functions</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Filtering</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>Smoothening</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>Sharpening</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>Contrast Stretching</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Histogram Equalization</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>Binary Image Generation</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Edge Detection</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Embossing</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>
3. PROPOSED ALGORITHM

The few most efficient algorithms utilized for image processing have been implemented in Photo vista Image Processing tool, which have been mentioned and described in the following sections.

3.1 Sobel Edge Detection Algorithms

The Sobel-edge enhancement operation extracts all of the edges in an image, regardless of direction. It is implemented as the sum of two-directional edge enhancement operations. The resulting image appears as an omni directional outline of the objects in the original image. Contrast brightness regions become black, while changing brightness regions become highlighted.

a. Application

The Sobel-edge enhancement operation is used to produce the edges of the objects in an image. The edges are any sharp brightness transitions rising from black to white or falling from white to black. This operation is omnidirectional, which means that all white-to-black and black-to-white edge transitions are highlighted, regardless of their direction in the image. The Sobel operation is more immune to image noise than the Laplacian operation, and it provides stronger edge discrimination.

b. Implementation

1. Define spatial convolution mask number 1
2. Pixel group process
3. Define spatial convolution mask number 2
4. Pixel group process
5. Dual-image pixel point process – addition

The Sobel-edge enhancement operation can produce results that are less than 0 or greater than 255. In these underflow and overflow cases, the resulting values are forced to 0 or 255, whichever is closest.

c. Algorithm

Procedure modified_sobel ( )
For y image(min_row) to image(max_row) do
For x image(min_column) to image(max_column) do
O(x,y) = 0
For mask_row = -1 to 1 do
For mask_column = -1 to 1 do
O(x,y) = O(x,y) + I(x,y) * mask_row * mask_column
End for
End for
Slope = sqrt(O(x,y)^2 + (O(x,y))^2)
If slope > 128 then O(x,y) = 0 else O(x,y) = 255 end if
End for
End for
End procedure

3.2 Smoothening: Low Pass Filtering

Low-Pass filtering smoothen an image by attenuating high-spatial-frequency details. The convolution mask weighting coefficients are selected to vary the cut-off point where higher frequencies become attenuated. Further, the resulting low-pass filtered image can be brightness-scaled down and summed with the original image to create milder low-pass filter effects.

a. Application

A low-pass filter can be used to attenuate image noise that is composed primarily of high-frequencies components such as impulse spikes and wide-band interference. The characteristics of low-frequency objects of interest can be enhanced by removing high-frequency image information using low-pass filtering. In particular, edges and sharp lines are smoothened, while low-frequency attributes are left untouched. For example, an image of an object containing only low spatial frequencies with a distracting high-frequency grid pattern can be enhanced in this way. A case like this is a biological specimen imaged under a microscope with measurement reticles. The low-pass effect is to blur the sharp lines of the reticles while leaving the low-frequency object relatively unaffected.

b. Implementation

1. Define spatial convolution mask
2. Pixel group process

In practice, low-pass mask coefficients are generally normalized to integer values. Then a brightness divider is applied to the resulting output value to bring the result within a 0 to 255 brightness range. The above masks can be converted to integer values with the appropriate brightness dividers. For more smoothing or higher attenuation of high frequencies 5x5 mask can be used instead of the 3x3 mask.

The algorithm to implement mask smoothing method using 3x3 mask block at the edges and 5x5 mask block for the inner pixels.

c. Algorithm

Procedure smoothing (mask, image, I)
For y image(min_row) to image(max_row) do
For x image(min_column) to image(max_column) do
O(x,y) = 0
For mask_row = -2 to 2 do
For mask_column = -2 to 2 do
O(x,y) = O(x,y) + I(x,y) * mask_row * mask_column
End for
End procedure
If \(x > \text{min}_\text{row}\) and \(x < \text{max}_\text{row}\) and \(y > \text{min}_\text{column}\) and \(y < \text{max}_\text{column}\) then

\[O(x,y) \leftarrow O(x,y) + I(x+\text{mask}_\text{row},y+\text{mask}_\text{column})\]

End for
End for

3.3 Median Filter

The low-pass filter is not so effective in noise reduction especially when the noise is random. It only dilates the noise and at the same time blurs the details and other sharp edges. A special type of low-pass filter is the median filter. The Median filter removes noise and detail by median-averaging the colors of adjacent pixels in the image. This filter determines the median value of neighboring pixels to smooth the image. This filter is particularly effective when the noise pattern consists of strong, spike-like components, and edge sharpness is to be preserved.

a. Application

Noise Reduction

b. Implementation

The median filter does not require convolution. It selects a window and finds the median of intensity of all the pixels inside that window. This median value is used to smooth in the noise with the background. The median algorithm requires a sorting sub-algorithm to sort the intensity values before finding the median of the sorted sequence.

c. Algorithm

Procedure median ( I,\text{median}_\text{extent})
n \leftarrow \text{median}_\text{extent}
Declare array buffer of size \(n\)
For \(y \leftarrow \text{image (min}_\text{column})\) to \(\text{image (max}_\text{column})\)
do
For \(x \leftarrow \text{image (min}_\text{row})\) to \(\text{image (max}_\text{row})\)
do
For \(i \leftarrow 0\) to \(n-1\) do
For \(j \leftarrow 0\) to \(n-1\) do
If \(x+j-n/2 \geq \text{image (min}_\text{column})\) and \(x+j-n/2 \leq \text{image (max}_\text{column})\) and \(y+i-N/2 \geq \text{image (min}_\text{row})\) and \(y+i-N/2 \leq \text{image (max}_\text{row})\) then
Buffer \((i+j) \leftarrow I(x+j-n/2,y+i-N/2)\)
End if
End for
End for
End case
End for
End for
End procedure

Procedure bubble_sort (buffer)
End if
End for
End for
bubble_sort (buffer)
End procedure

First the pixels in the window of size \(n\) in stored in an array buffer. Then the intensity values in the array are sorted in the ascending order. Here we have sorted by the bubble sort method. The bubble_sort procedure takes the array buffer as arguments and after sorting puts the sorted values in the array. Any other sorting method like quick-sort can also be used. Then the median value of the values in the sorted array is obtained and that is the resultant intensity value of the center pixel in the window. This process is repeated for each pixel by placing the pixel to be operated on, at the center of the window. The window size \(n\) determines the extent of median cutting or noise reduction. The value of \text{median}_\text{extent} gives the window size in generally an odd number. Since a sorting operation has to be done for each pixel, the median operation is a bit slower than other algorithms. Higher the value of \(n\) (or \text{median}_\text{extent}), more values would have to be sorted (\(n^2\)) and so slower will be the operation.

4. EXPERIMENTAL RESULTS

Proposed tool is developed using Java as Programming Language. The following output results are seen after processing the input images with few options of Photo vista Image Processing Tool:

![Fig 1: Original Image](image-url)
5. CONCLUSION & FUTURE RESEARCH DIRECTION

In this paper we have discussed various image processing tools and presented an integrated tool to perform the same. Experimental results show that the proposed tool has most of the options that the user needs for pre-processing the images before they can be used.

Low and high level features captured can be used for implementing Content Based Image Retrieval in future. Content Based Image Retrieval with data mining can produce better results for searching images which will be useful for the faster access of multimedia, bioinformatics, medical and biological databases and retrieval of criminal face, finger databases from the criminal detection system.

REFERENCES


[10] Yan-Min Chen; Xiao-Long Wang; Bing-Quan Liu; In proceedings of International Conference on Multi-
document summarization based on lexical chains
Machine Learning and Cybernetics,
Volume 3, Aug. 2005, 1937 - 1942

“Robust document image understanding technologies”,


