Non Linear Image Enhancement

SAIYAM TAKKAR
Jaypee University of information technology, 2013

SIMANDEEP SINGH
Jaypee University of information technology, 2013

Abstract

An image enhancement algorithm based on a neighborhood dependent nonlinear model is presented to improve visual quality of digital images.

In our research, we are using various non linear digital enhancement techniques so as to remove noise in a digital image as well as to improve the visual quality of digital images as well as images that exhibits dark shadows due to limited dynamic range of imaging.

We have used nonlinear image enhancement because nonlinear image enhancement tools are less susceptible to noise. Noise is always present due to the physical randomness of image acquisition systems. For example, underexposure and low-light conditions in analog photography conditions lead to images with film-grain noise which, together with the image signal itself, are captured during the digitization process.

Nonlinear methods effectively preserve edges and details of images while methods using linear operators tend to blur and distort them.

We have developed a GUI in which we have used mean filtering non linearly to enhance an image which is able to enhance the luminance in the dark shadows keeping the overall tonality consistent with that of the input image

DIGITAL IMAGE

An image is a two-dimensional function f(x,y), where x and y are the spatial (plane) coordinates, and the amplitude of f at any pair of coordinates (x , y) is called the intensity of the image at that level. If (x , y) and the amplitude values of f are finite and discrete quantities, we call the image a digital image. A digital image is composed of a finite number of elements called pixels, each of which has a particular location and value of f at (x, y) is proportional to the brightness of the image at that point. It is represented by a 2-dimensional integer array. The digitized brightness value is called the gray level value. Each element of the array is called a pixel. A digital N×N image looks like this:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,N-1) \end{bmatrix}$$

Figure1.1

DIGITAL IMAGE ENHANCEMENT

The principal objective of enhancement is to process an image so that the result is more suitable than the original image for a specific application.Image enhancement tools are often classified into (a) **point operations** and (b) **spatial operations**. Point operations include contrast stretching, noise clipping, histogram modification, and pseudo-coloring. Point operations are, in general, simple nonlinear operations that are well known in the image processing literature.

Spatial operations used in image processing today are, on the other hand, typically linear operations. The reason for this is that spatial linear operations are simple and easily implemented. Although linear image enhancement tools are often adequate in many applications, significant advantages in image enhancement can be attained if nonlinear techniques are applied. Nonlinear methods effectively preserve edges and details of images while methods using linear operators tend to blur and distort them.
Image enhancement approaches fall into two broad Categories: Spatial domain methods and Frequency domain method

SPATIAL DOMAIN METHODS

spatial domain refers to the aggregate of pixels composing an image. Spatial domain methods are procedures that operate directly on the pixels

g(x,y) = T[f(x,y)]

f(x, y) : input image

g(x ,y) : output image

T [ ] : an operator defined over some neighbourhood of (x,y)

The principal approach in defining a neighbourhood about a point (x,y) is to use a square or rectangular sub image area centred at (x,y).

The centre of the subimage is moved from pixel to pixel starting at the top left corner. The operator T is applied at each location (x,y) to yield the output g, at that location.

Image Averaging

Image noise can compromise the level of detail in your digital or film photos, and so reducing this noise can greatly enhance your final image or print. The problem is that most techniques to reduce or remove noise always end up softening the image as well. Some softening may be acceptable for images consisting primarily of smooth water or skies, but foliage in landscapes can suffer with even conservative attempts to reduce noise.
Image averaging works on the assumption that the noise in your image is truly random. This way, random fluctuations above and below actual image data will gradually even out as one averages more and more images. If you were to take two shots of a smooth gray patch, using the same camera settings and under identical conditions (temperature, lighting, etc.), then you would obtain images similar to those shown on the left.
Image averaging is common in high-end astrophotography, but is arguably underutilized for other types of low-light and night photography. Averaging has the power to reduce noise without compromising detail, because it actually increases the signal to noise ratio (SNR) of your image. An added bonus is that averaging may also increase the bit depth of your image — beyond what would be possible with a single image.

Log Transformation:

Image enhancement simply means, transforming an image *f* into image *g* using *T*. Where *T* is the transformation. The values of pixels in images *f* and *g* are denoted by *r* and *s*, respectively. As said, the pixel values *r* and *s* are related by the expression,

$$s = T(r)$$

where *T* is a transformation that maps a pixel value *r* into a pixel value *s*. The results of this transformation are mapped into the grey sclale range as we are dealing here only with grey scale digital images. So, the results are mapped back into the range [0, L-1], where $L=2^k$, k being the number of bits in the image being considered. So, for instance, for an 8-bit image the range of pixel values will be [0, 255].

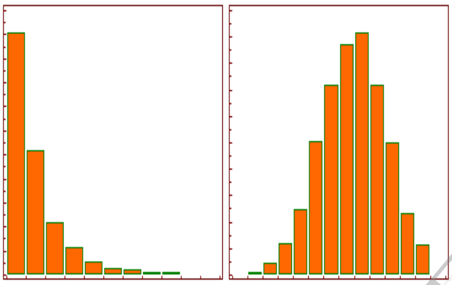
The log transformation is given by the expression,

$$s = c \log(1 + r)$$

where *c* is a constant and it is assumed that *r*≥0. The shape of the log curve in fig. A tells that this transformation maps a narrow range of low-level grey scale intensities into a wider range of output values. And similarly maps the wide range of high-level grey scale intensities into a narrow range of high level output values. The opposite of this applies for inverse-log transform. This transform is used to expand values of dark pixels and compress values of bright pixels.

The logarithm function tends to squeeze together the larger values in your data set and stretches out the smaller values.

The following illustration shows the histogram of a log-normal distribution (left side) and the histogram after logarithmic transformation (right side).



When you select log transformation, MedCalc computes the base-10 logarithm of each data value and then analyses the resulting data. For ease of interpretation, the results of calculations and tests are backtransformed to their original scale.

Original number = x

Transformed number x'=log$_{10}$(x)

Bit Plane Slicing

Instead of highlighting gray level images, highlighting the contribution made to total image appearance by specific bits might be desired. Suppose that each pixel in an image is represented by 8 bits. Imagine the image is composed of 8, 1-bit planes ranging from bit plane1-0 (LSB)to bit plane 7 (MSB).

In terms of 8-bits bytes, plane 0 contains all lowest order bits in the bytes comprising the pixels in the image and plane 7 contains all high order bits.

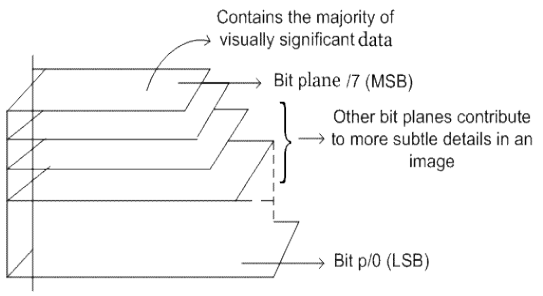

Contains the majority of visually significant data
Bit plane /7 (MSB)
Other bit planes contribute to more subtle details in an image
Bit p/0 (LSB)

Figure (5.15)

Separating a digital image into its bit planes is useful for analyzing the relative importance played by each bit of the image, implying, it determines the adequacy of numbers of bits used to quantize each pixel , useful for image compression

.In terms of bit-plane extraction for a 8-bit image, it is seen that binary image for bit plane 7 is obtained by proceeding the input image with a thresholding gray-level transformation function that maps all levels between 0 and 127 to one level and maps all levels from 129 to 253 to another (eg. 255).

Gray Level Slicing

Gray-scale modification (also called gray-level scaling) methods belong in the category of point operations and function by changing the pixel's (gray-level) values by a mapping equation. The mapping equation is typically linear (nonlinear equations can be modeled by piecewise linear models) and maps the original gray-level values to other, specified values. Typical applications include contrast enhancement and feature enhancement.

The primary operations applied to the gray scale of an image are to compress or stretch it. We typically compress gray-level ranges that are of little interest to us and stretch the gray-level ranges where we desire more information. This is illustrated in Figure 4.2-1a, where the original image data are shown on the horizontal axis and the modified values are shown on the vertical axis. The linear equations corresponding to the lines shown on the graph represent the mapping equations. If the slope of the line is between zero and one, this is called gray-level compression, whereas if the slope is greater than one, it is called gray-level stretching.

In Figure, the range of graylevel values from 28 to 75 is stretched, while the other gray values are left alone. The original and modified images are shown in Figures, where we can see that stretching this range exposed previously hidden visual information. In some cases we may want to stretch a specific range of gray levels, while clipping the values at the low and high ends.

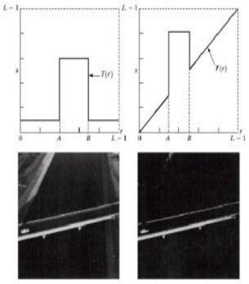


Image subtraction

The pixel subtraction operator takes two images as input and produces as output a third image whose pixel values are simply those of the first image minus the corresponding pixel values from the second image. It is also often possible to just use a single image as input and subtract a constant value from all the pixels. Some versions of the operator will just output the absolute difference between pixel values, rather than the straightforward signed output.

How It Works

The subtraction of two images is performed straightforwardly in a single pass. The output pixel values are given by:

$$Q = P1(I , j) – P2(I , j)$$

Or if the operator computes absolute differences between the two input images then:

$$Q = |P1(I , j) – P2(I , j)|$$

Or if it is simply desired to subtract a constant value *C* from a single image then:

$$Q = P1(I , j) – c$$

If the pixel values in the input images are actually vectors rather than scalar values (*e.g.* for color images) then the individual components (*e.g.* red, blue and green components) are simply subtracted separately to produce the output value.

Implementations of the operator vary as to what they do if the output pixel values are negative. Some work with image formats that support negatively-valued pixels, in which case the negative values are fine (and the way in which they are displayed will be determined by the display color map). If the image format does not support negative numbers then often such pixels are just set to zero (*i.e.* black typically). Alternatively, the operator may `wrap' negative values, so that for instance -30 appears in the output as 226 (assuming 8-bit pixel values).

If the operator calculates absolute differences and the two input images use the same pixel value type, then it is impossible for the output pixel values to be outside the range that may be represented by the input pixel type and so this problem does not arise. This is one good reason for using absolute differences.

FREQUENCY DOMAIN METHODS

•Fourier Series: Any function that periodically repeats itself can be expressed as the sum of sines/cosines of different frequencies, each multiplied with a different coefficient.

•Fourier Transform: Functions that are not periodic, whose area under the curve is finite, can be expressed as the integral of sines and/cosines multiplied by a weighting function.

Spatial Filtering

Neighbourhood operations use a subimage that has the same dimension as the neighbourhood. The subimage is called a filter, mask, kernel, template, or window. The values in filter subimage are referred to as coefficients, rather than pixels. The response of the filter at a point is calculated using a predefined relationship. For linear spatial filtering, the response is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask.

Smoothing Spatial Filters

Smoothing Filters are used for blurring and for noise reduction. Blurring is used for removal of small details prior to object extraction and Bridging of small gaps in lines or curves. Smoothing Linear Filters (Averaging Filters) replace the average value defined by the filter mask and have the undesirable effect of blur edges.

Order-Statistics Filters

Order statistics filters are Nonlinear spatial filters. Response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter. Replacing of the centre pixel with the value determined by the ranking result. Median filter replaces the pixel value by the median value in the neighbourhood. Excellent noise-reduction capabilities with less blurring than linear smoothing filters. Effective for impulse noise (salt-and-pepper noise)

Sharpening Filters

The principal objective of sharpening is to highlight fine detail in an image or to enhance detail that has been blurred. While averaging is analogous to integration, sharpening could be accomplished by spatial differentiation. Image differentiation enhances edges and other discontinuities

First- and second-order derivatives

- First-order derivatives produce thicker edges in an image and stronger response to a gray-level step. Another feature is Edge extraction

- Second-order derivatives produce double response at step changes in gray level and stronger response to fine detail

Noises in an Image

Digital images are prone to a variety of types of noise. Noise is the result of errors in the image acquisition process that result in pixel values that do not reflect the true intensities of the real scene. There are several ways that noise can be introduced into an image, depending on how the image is created. For example:If the image is scanned from a photograph made on film, the film grain is a source of noise. Noise can also be the result of damage to the film, or be introduced by the scanner itself.If the image is acquired directly in a digital format, the mechanism for gathering the data (such as a CCD detector) can introduce noise.Electronic transmission of image data can introduce noise.

Various Types of noises

1. Gaussian Noise
2. Rayleigh Noise
3. Salt &Pepper Noise

Gaussian Noise

Gaussian noise is properly defined as the noise with a Gaussian amplitude distribution. This says nothing of the correlation of the noise in time or of the spectral density of the noise. Labeling Gaussian noise as *white* describes the correlation of the noise. It is necessary to use the term **white Gaussian noise** to be precise. Gaussian noise is sometimes equated to white Gaussian noise, but this is not necessarily the case.

Rayleigh Noise

Rayleigh noise is the noise described by the random process with the Rayleigh distribution of probability density function.

$$W\left(\frac{u}{\sigma_n}\right) = \frac{u}{\sigma_n}\exp\left(\frac{u^2}{2\sigma^2}\right), u \geq 0$$

$$= 0, u < 0$$

Where u is instantaneous noise voltage

σ_n^2 is noise variance.

Salt & Pepper Noise

An image containing salt-and-pepper noise will have dark pixels in bright regions and bright pixels in dark regions. This type of noise can be caused by analog-to-digital converter errors, bit errors in transmission, etc.

For an 8-bit image, the typical value for pepper noise is 0, and 255 for salt-noise.

METHODOLOGY

We have worked on pattern recognition of Braille language. We have developed an algorithm which helps to remove misalignment of Braille which occurs during manual scanning. During our research we found various linear enhancement techniques and the main drawback of linear enhancement technique is that some vital information is lost because it works on that part of the image also that doesn't need to be enhanced. So to avoid this we chose to work on non linear methods.
So far

We have used a noisy image which is corrupted with 'Salt & Pepper' noise. We have first created an image containing 'Salt &Pepper' noise using MATLAB R2008a . We have further enhanced the image using java. We have developed an algorithm which enhances the image in a non linear way.
Image Library of Java has been used to read the image.

GUI (Graphic User Interface) has been designed in Java.

PREVIOUS ALGORITHM

1. Read the image.
2. A Matrix of same order as that of image is generated.
3. The Matrix generated represents all the pixel values of the image to be enhanced.
4. We have given particular conditions according to which an image needs to be enhanced.
5. We have designed a Graphic User Interface in JAVA.
6. The interface allows user to choose the conditions according to his needs.
7. We further check each value of the obtained matrix.
8. Noise is detected by the further procedure.
9. Take the difference of 2 adjacent pixels and check the corresponding difference according to the given condition by the user.
10. If the difference satisfies the given conditions then store the pixel location in an array.
11. Apply mean filtering technique to that particular pixels only keeping rest of the image intact.
12. Substitute the noise which is present in the original image by the values calculated by the mean filtering.
13. By using this algorithm, we only work on those pixels which contain noise.
14. Main advantage of this algorithm is we do not loose information of an image as pixels containing noise is only affected

IMPROVED METHOD

Flow Chart

The following flow chart describes the algorithm we have used to Non Linearly Enhance an Image.

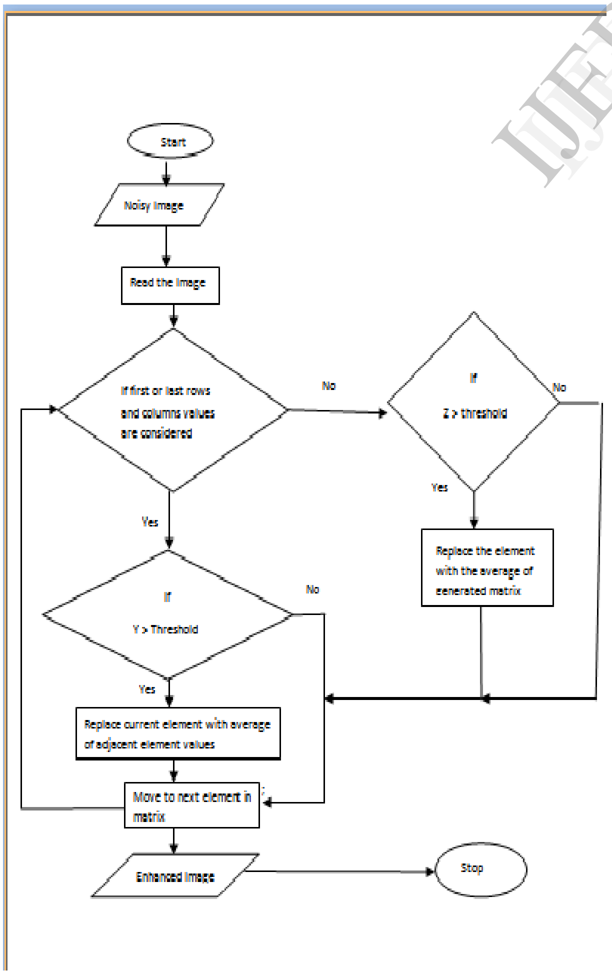
We read the gray image and generated a matrix of the same order as that of the image.

Terms used:

Y = difference of adjacent elements in the matrix.

Z = difference of the element with all the neighboring elements in the 3 X 3 matrix generated by considering the element to be processed as the center of the matrix.

Threshold = Input taken from the user.


Start
Noisy Image
Read the image
If first or last rows and columns values are considered
No
Yes
If
Z > threshold
No
Yes
Replace the element with the average of generated matrix
If
Y > Threshold
No
Yes
Replace current element with average of adjacent element values
Move to next element in matrix
Enhanced Image
Stop

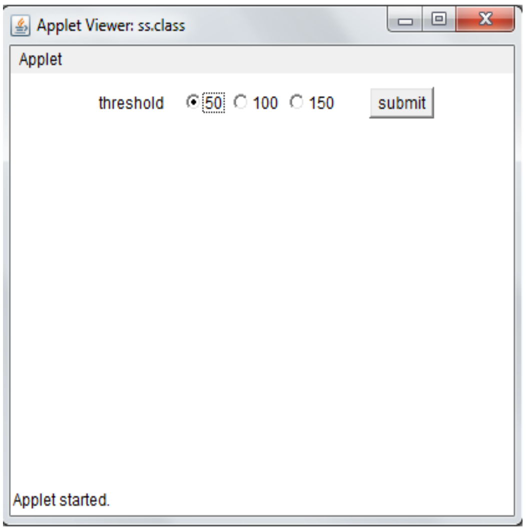
Algorithm

- We have designed a Graphic User Interface in JAVA.
- The interface allows user to choose the conditions according to his needs.
- First check for the first and last row's and column's.
- Take the difference of adjacent pixels and check the corresponding difference according to the given condition by the user.
- If the difference satisfies the given conditions then store the pixel location in an array.
- Apply mean filtering technique to that particular pixels only keeping rest of the image intact.
- Substitute the noise which is present in the original image by the values calculated by the mean filtering.
- Now consider the rest elements of the matrix , compare the respective element with every other element of a 3X3 matrix and if the difference from any other element is greater than user defined threshold .
- Replace the current element value with the average of all the elements of the 3X3 matrix.

Main advantage of this algorithm is we do not loose information of an image as pixels containing noise is only affected.

RESULT

Below we have shown the developed Graphic User Interface.

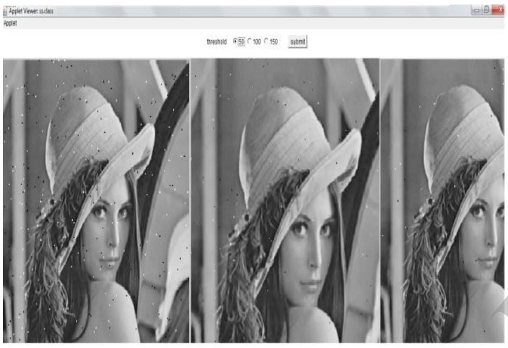


Applet Viewer: ss.class

Applet

threshold ● 50 ○ 100 ○ 150 submit

Applet started.



ORIGNAL IMAGE OUTPUT IMAGE 1 OUTPUT IMAGE 2

(FIRST METHOD) (IMPROVED METHOD)

Further in GUI we compare the original image with the 2 enhanced images, one enhanced using median filter and the other one enhanced using our proposed algorithm. We are able to easily see the affect of our algorithm, as image obtained using our algorithm is better than the original image.

We have been successful in removing some extent of the noise (Salt & Pepper) but still some noise is present in the image, but in the overall sense no important information has been lost.

Refrences

- http://www.ece.rice.edu/~wakin/images/
- Digital Image Processing 2nd Edition –R Gonzalez , R Woods
- http://www.cambridgeincolour.com/tutorials/gamma-correction.htm
- http://www.cvip.uofl.edu
- http://www.mathworks.in
- Mr. Mohammed Ghouse,Dr. m. siddappa "adaptive techniques based high detection and reduction of a digital image" journal of theoretical and applied information technology© 2005 - 2011 jatit & lls. all rights reserved
- Mr. Ankur. N. Shah, Dr. K. H. Wandra," Introduction to noise, image restoration and comparison of various methods of image restoration by removing noise from image", Volume : 2 | Issue : 1 | October 2012 | ISSN - 2249-555X