**STUDY OF QUALITY AND RANDOMNESS PRODUCED BY INTENSITY TRANSFORMATION**

*A Project Submitted in Internship Program*

*(From 1stJune to 30thJune, 2018)*

*By*

***Akshay Kumar***

*Roll No*. – 15505

*of*

*Bhagalpur College of Engineering, Bhagalpur*

*Under The Guidance of*

**Prof. Tamal Pal**

**(Assistant Professor, Computer Science And Technology Department )**

****

**DEPARTMENT OF COMPUTER SCIENCE & TECHNOLOGY**

**INDIAN INSTITUTE OF ENGINEERING SCIENCE & TECHNOLOGY, SHIBPUR, Howrah-711103**

****

**DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY**

**INDIAN INSTITUTE OF ENGINEERING SCIENCE & TECHNOLOGY, SHIBPUR**

**HOWRAH- 711103**

**CERTIFICATE**

This is to certify that the project entitled **“Study of Quality and Randomness produced by Intensity Transformation”** submitted by **Akshay Kumar (Roll: 15107)** of Bhagalpur College of Engineering, Bhagalpur, under my guidance and supervision of internship program **(from 1st June to 30th June, 2018)** in the Department of Computer Science & Technology, Indian Institute of Engineering Science & Technology, **(**Formerly Bengal Engineering and Science University) Shibpur.

Date:

..................................................

**(Prof. Tamal Pal)**

Assistant Professor

Department of Computer Science & Technology

**COUNTERSIGNED BY:**

………………………………

**(Prof. Sulata Mitra)**

Professor & Head,

Department of Computer Science & Technology

**Acknowledgement**

It is a pleasure to acknowledge many people who knowingly and unwittingly helped us, to complete our Summer Internship. First of all let us thank God for all the blessings, which carried us through all these years.

I express my gratitude to **Prof. Tamal Pal, Professor, Department of Computer Science And Technology, IIEST Shibpur** for providing this opportunity of getting one month internship training, and also for his valuable suggestions, active supervision and constant encouragement.

.

Lastly I would like to thank to other faculty members of **Computer Science And Technology.**

..................................

Akshay Kumar

Date: 28/06/2018

Department of Computer Science And Engineering ,

Bhagalpur College Of Engineering, Bhagalpur.

Roll No. – 15505

Email :- akshaygugly123@gmail.com

**Contents**

**Table of content PageNo**

1. **Introduction ................................................................... 1**
2. **Overview ................................................................... 1**
3. **Approach .......................................................... 7**
4. **Experimental Results ......................................................... 10**
5. **Conclusion ........................................................... 16**
6. **References ........................................................... 16**
7. **Appendix ........................................................... 18**

Image Transformation using 2D and 3D curves

1. ***Introduction:-***

My intended purposed is to transform the images using various equations and to see how much randomness it is producing and effect on the quality of the image due to transformation. These things are used in the security purpose to modify the image such a way that it can produce a random image and then there should be some provision to get the original image back. Thus we can secure the Information containing in the image.

1. ***Overview:-***

Image Transformation is a technique by which we can explore the information in the image by modifying some of its characteristics. Here we are using image intensities and then modifying the intensity to get some transformation.

So the basic idea here is to take a matrix which is containing intensities of the image and then apply the transformation functions (It can be 2D or 3D) to get the transformed intensity matrix. Now calculate randomness of image intensities produced, check quality, or save for the reconstruction work. Use Transformed matrix to reconstruct the image. And then check the randomness and quality.

Here is the overview of all the functions that I have used:-

***2.1 2D Transformation functions :-***

**2.1.1 Negative Transformation (Linear):**

The easiest function to apply on image, the basic idea is to reverse the value where the lowest will change into highest intensity and the highest will change into the lowest intensity and the rest will change accordingly.

The equation representing Negative transformation is **f(x) = L-1-x**, where L is the highest intensity in that scale and x is input intensity.

For grayscale images L=256. This transformation is used when we have to see the area where things are darker to extract some information.

**Source :- ImageProcessing by Akshay Kumar\2D curves**

------------------------------------------------------

**2.1.2 Logarithmic Transformation (Log):**

Logarithmic Transformation is used to increase the contrast of lower intensity of the images. This transformation makes images looks brighter. And the brightness depends on the constant c.

The equation representing Log Transformation is **f(x) =c\*log (1+x),** where c is a constant which govern how bright the image will be and x is the input intensity. This is useful when we have to see thing which is no clear due to lower contrast and in lower intensity that is darker.

**Source :- ImageProcessing by Akshay Kumar\2D curves**

-----------------------------------------------------

**2.1.3 Gamma Transformation (power):**

Gamma Transformation can brighten the intensity or darken the intensity. So the brightening or darkening depend on the value of gamma (the exponent) which can increase (or decrease) the intensity and thus the resultant image will be brighter (or darker).

The equation representing this transformation is **f(x)=c\*(x)^(gamma),** where c is a constant used for keeping the value of intensity in some range and gamma is used to control the curve that is the intensity.

**Source :- ImageProcessing by Akshay Kumar\2D curves**

-----------------------------------------------------

**2.1.4 Contrast-Stretching Transformation:**

Basically the contrast is the difference of the maximum intensity to the minimum intensity. So Contrast-Stretching Transformation increases the contrast. So it stretch the histogram to fit in the intensity domain. It will darken the darker and brighten up the brighter thus increasing the contrast of the image.

The equation representation is **f(x)= 1/(1+(m/(x + eps ))^E)** , where m is the mid-line the threshold value from where the transformation in either side is happening, eps is a constant which is distance between two doubles used to prevent divide by zero, E is used to control the slope of the function that is the intensity value.

**Source :- ImageProcessing by Akshay Kumar\2D curves**

***2.2 3D Transformation functions:-***

**2.2.1 Ellipsoid Transformation Function:-**

The general equation of an ellipsoid is

http://tutorial.math.lamar.edu/Classes/CalcIII/QuadricSurfaces_files/eq0003M.gif

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**2.2.2 Cone Transformation Function:-**

The general equation of a cone is

http://tutorial.math.lamar.edu/Classes/CalcIII/QuadricSurfaces_files/empty.gif http://tutorial.math.lamar.edu/Classes/CalcIII/QuadricSurfaces_files/eq0005M.gif

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**2.2.3 Hyperboloid-One-Sheet Transformation Function:-**

The general equation of a hyperboloid in one sheet is

http://tutorial.math.lamar.edu/Classes/CalcIII/QuadricSurfaces_files/eq0010M.gif

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**2.2.4 Hyperboloid-Two-Sheet Transformation Function:-**

The general equation of hyperboloid in two sheets is

http://tutorial.math.lamar.edu/Classes/CalcIII/QuadricSurfaces_files/eq0011M.gif

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**2.2.5 Elliptic-Paraboloid Transformation Function:-**

The general equation for elliptic paraboloid is

http://tutorial.math.lamar.edu/Classes/CalcIII/QuadricSurfaces_files/eq0012M.gif

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**2.2.6 Hyperbolic Paraboloid Transformation Function:-**

The general equation for Hyperbolic Paraboloid is

http://tutorial.math.lamar.edu/Classes/CalcIII/QuadricSurfaces_files/eq0014M.gif

**So,** we used this function in 2d manner by making z a constant. When z becomes constant then whole equation is transformed in only x and y which is a 2d function. Now this function is used for the transformation.

We divide z into some discrete value and then using those discrete values a family of 2d equation is obtained which is then used in the transformation. I have used Z = [ -1 1 3 5 7 ] for transformation and Z=[-3 -1 1 3 5 7] for the reconstruction work.

After getting 2d equations, x is treated as input and y is treated as output during transformation, while y is treated as input and x is treated as output during reconstruction work.

F(input)=output, where F() is the 2d equation obtained after making z constant and choosing appropriate input output over x and y.

**Source :- ImageProcessing by Akshay Kumar\3D curves**

***2.3 Randomness Test (Frequency test):-***

**2.3.1 Frequency (Mono-bit) Test:-**

The basic idea is that the number of zeroes and ones expected to be same in any random sequence of bit generated by any random number generator (RNG) or pseudo random number generator(PRNG).

The length of the string is represented by n and the sequence of bit is represented by E=E1,E2,E3.....En;. Now the test is calculated as test statistic Sobs=|Sn|/sqrt(n), where Sn =(no\_of\_ones - no\_of\_zeroes). The reference distribution for the test statistic is half normal distribution X=Sobs/sqrt(2);

Now pvalue=1-erfc(X), where erfc is complemented error function .

Decision rule is made at 1% level that is 0.01 value, if p-value is <0.01 then conclude that the sequence is non-random, and if p-value is >= 0.01 then conclude the sequence is random.

**Source :- ImageProcessing by Akshay Kumar\3D curves**

***2.4 Reverse Of Transformation Functions:-***

The reverse of Transformation function is just changing the previous input as output and previous output as input to create a new inverse or reverse function.

Lets say we have f(x)=y, then we will convert this function in the form g(y)=x, for example if we have y=x+1, then we can say that f(x)=x+1=y , and when we apply reverse we will get g(y)=y-1=f(x)-1=x , for that particular function in x and y.

So, we have applied same this to all the transformation function that I have used to construct the reverse Transformation function which will be used for the reconstruction of image.

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**Source :- ImageProcessing by Akshay Kumar\2D curves**

***2.5 PSNR:-***

peak signal-to-noise ratio (PSNR) is the ratio between the maximum possible pixel of a image and the pixel of corrupting noise that affect the fidelity of its representation. It is expressed in terms of the logarithmic decibel scale

Psnr defined via mse(mean squared error) where mse=sum((I(:)-K(:)).^2)/numel(I); and mse is mean squared error , numel is matlab function which return no of elements. Psnr is defined as psnr=20.log10(Maxi)-10log10(mse);

Since we are converting our original image using im2double and making the transformation the pixel values will be in 0 to 1 thus Maxi=1; and psnr=-10log10(mse) if mse is zero the psnr==INF means two images are identical, if mse is inf the psnr==-INF means we cannot compute the psnr as the reconstructed matrix contains infinity

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**Source :- ImageProcessing by Akshay Kumar\2D curves**

***2.5 SSIM:-***

SSIM (Structural similarity) is calculated over various windows of an image but I am considering whole image as a window.

ssim(X,Y)=((2\*mean\_X\*mean\_Y+c1)(2\*covariance\_x\_y+c2))/((mean\_x^2+mean\_y^2+c1)(variance\_x^2+variance\_y^2+c2));

mean\_x=mean(X(:));

mean\_y=mean(Y(:));

variance\_x=var(X(:));

variance\_y=var(Y(:));

covariance\_x\_y=sum((X(:)-mean\_x).\*(Y(:)-mean\_y))/numel(X);

c1=(k1\*L)^2 c2=(k2\*L)^2

k1=0.01 k2=0.03 and L= dynamic range of pixel values

**Source :- ImageProcessing by Akshay Kumar\3D curves**

**Source :- ImageProcessing by Akshay Kumar\2D curves**

1. ***Approach :-***

***3.1 Calculating randomness for transformation***

First image is transformed into a intensity matrix using matlab function imread(). Let say we got a matrix im which is the intensity matrix, Now we choose any transformation function with their with their required parameter and the matrix im is transformed. Every transformed function will return an transformed matrix. So, say that matrix is X, if there is a chance of getting an imaginary element in the transformed matrix then apply abs() to consider only absolute value ( often seen while applying 3-d curves). Now calculated the absolute difference in the original intensity matrix and transformed matrix and call this difference of matrixes as diff. Now using median transform this matrix into 0’s and 1’s and then supply to the randomness test function, in my case frequency test for checking randomness.

**Steps for calculating Randomness**

calculateRandomness(){

im=imread(); //Read Image

im=im2double(im); //change into double

X=any\_transformation\_function\_2d\_3d(im);

// apply transformation

diff=abs(im-abs(x));// diff of two matrix

convMat= MatrixConversion(diff);

pvalue=FrequencyTest(convMat);

}

MatrixConversion( im){

median=find\_median\_of\_im;

for\_each element x in im

if x<median

convert x to 0;

if x>=median

convert x to 1;

return im;

}

FrequencyTest ( convMat ){

l=numel(convMat); // total no of elements in matrix

calculate Total no. of one

calculate total no of zeroes

sn=abs(Total\_no\_of\_ones-total\_No\_of\_zeors);

sobs=sn/sqrt(l);

X=sobs/sqrt(2);

pvalue=q-erfc(X);

return pvalue;

}

***3.2 Reconstruction of image***

What I have done is for test purpose I first applied the transformation to the original matrix and then saved the transformed matrix into a csv file ( in case of 3d curve two csv file one for real part of the intensity and second for the imaginary part of the intensity ). Then I have read the original image using imread() applied transformed it into double using im2double, then read csv file to get the transformed intensities and the applied reverse of the transformation with different parametric value to get a reconstructed image and calculated the psnr and ssim test to see how to quality does reconstructed image have with respect to original image. Reading and writing in csv are explained later in this section.

***3.3 Calculating psnr for transformation***

Psnr is calculated over two matrixes either between the original and reconstructed to see how much quality is lost, or between the transformed image and the original image to see how much quality of the original image is left in the transformed image.

To calculate psnr between the transformed and original image, read a image using imread() and save it into im, then apply transformation function to get transformed image X. Then fed these two matrix into the mypsnr() function to calculate psnr( whenever there is a chance of complex no apply abs before passing the matrix to run calculation smoothly ).

To calculated psnr between original image and the reconstructed image take the original intensity matrix and reconstructed image intensity matrix and pass them to mypsnr() function.

***3.4 Calculating ssim for transformation***

The procedure to calculate ssim is same as that of psnr except for the function that I used is myssim() to calculated ssim. And I am considering whole image as the window, so the size of both the matrix should be same.

**Steps to calculate image Quality**

[psnr, ssim] imageQuality() {

orig=imread(‘image\_name’); //read original image

orig=im2double(im); //change it into double

X=getTransformedMatrixFromCsv(‘test\_name’);

reconstructedImg = Reverse\_transformation(X);

psnr = mypsnr (orig, reconstructedImg);

ssim = myssim (orig, reconstructedImge);

}

[ X ] getTransformedMatrixFromCsv( test\_name){

mat=csvread(test\_name);

first\_element=mat(1,1);

if first\_element ==3

transform the mat into n\_by\_m\_by\_3

return mat;

}

[ psnr ] mypsnr ( orig, reconstructed ){

calculate mse(orig, reconstructed );

psnr = calculate using the formula

}

[ssim] myssim(orig, reconstructed){

calculate mean of orig

calculate mean of reconstructed

calculate variance of orig

calculate variance of reconstructed

calculate covariance of orig and reconstructed

c1=0.03\*(max\_pixel)

c2=0.01\*( max\_pixel)

calculate the value of ssim using formula as described in ssim section

}

***3.5 Challenges in calculating transformation***

Various matrix operation like power and log does not apply to integers that why we have to transform it to double and that can be done in two way in matlab one is to use double() other is to use im2double(), im2double() will make all intensity to lie between 0 and 1 which can be again shown using imshow(), but double() does not do that, that is why, I have used im2double to transform intensity matrix into double.

Often in applying 3d curves for the transformation we get complex values which cannot be depicted as image by imshow() function that’s why I have used absolute values of the complex no ( applied abs() ) to show the image and carry out calculations.

***3.6 Generating test file***

First I have applied transformation function to get the transformed intensity matrix. Then saving that matrix using imwrite() was having a problem of loss of control over the values of intensity so I was having two options either encode decode the transformed matrix and save using imwrite() or to directly save the matrix using csvwrite(). So I have used csvwrite() function to save the matrix as test\_\_.csv in order to use it for the reconstruction. In the process of saving I have used first element of the matrix to store the third dimension of matrix because \_\_x\_\_x3 will get transformed into \_\_x\_\_ when saved as csv, so in order to know the dimension in future reference I have used first element.

Now in order to extract the transformed matrix what I am doing is reading the whole csv file and then checking if the first element is 3 or not if yes then break down the csv columns into 3 equal part and then transform it into three dimension.

In case of 3d curves, first add the real part to imaginary part element wise and then do the same thing as done above to get the transformed intensity matrix.

1. ***Experimental result:-***

**Results are in excel file in separate folder 2D curves and 3D curves of the link which I have shared in the reference section.**

* 2d curves
  1. randomnessandquality.xlsx
  2. randomnessvsthreshold.xlsx
  3. psnr and ssim of reconstructed image.xlsx
* 3d curves
  1. RandomnessAndQulaityofTransformedImage.xlsx
  2. RandomenssVsThresholdofTransformedImageusingSurvfacecurves.xlsx
  3. RandomnessAndQualityofReconstructedImage.xlsx

***4.1 2D transformation***

**4.1.1 Randomness vs quality** 

**4.1.2 Randomness vs Threshold (median value used to convert matrix into 0’s and 1’s)**

****

**4.1.3 Image quality of reconstructed image**



**4.2 3d Transformation**

**4.2.1 Randomness vs quality**

****

**4.2.2 Randomness vs Threshold**

****

**4.2.3 Randomness and quality of reconstructed image**

****

1. ***Conclusion:-***

* I have applied every standard intensities transformation function and also some 3d function by keeping their z axis constant to find out the quality and the randomness they produce.
* I found out that log transformation produces same transformation for all values of c.
* Negative transformation produces higher randomness of almost 1 .
* In terms of quality contrast stretching transformation produces good quality.
* For the randomness I have used different values of median and found out that if we change the median value slightly then the randomness increase towards 1.
* When doing the reconstruction of image we get almost the same image quality of the original image on seeing the value of psnr and ssim for the values of the parameter that we have used to create test cases on applying all the transformation with different parametric values.

1. ***References-***

* Used images in my experiment are random.
* For the calculation purpose I have used Matlab.
* For the definition of psnr and ssim I have used Wikipedia.
* For the frequency test and other functions I have used various online present materials.
* The link to downloads all the documents and matlab program used in the process is https://drive.google.com/open?id=1Ku5t30nL4iOiSfIn\_Ai6actAX\_ZVBjWW

Appendix