

**THE UNIVERSITY OF TEXAS AT ARLINGTON, TEXAS  
DEPARTMENT OF ELECTRICAL ENGINEERING**

**EE 5356**

**DIGITAL IMAGE PROCESSING**

**PROJECT # 2**

**by**

**SOUTRIK MAITI**

**1001569883**

**Presented to**

**Dr. K.R.RAO**

**Feb 15, 2019**

1. UNIFORM QUANTIZER

*The code has been written according to the pseudocode provided. Comments are added in the MATLAB script to explain the algorithm*

*MATLAB CODE:*

%%MATLAB implementation of Uniform Quantizer

%%Read the image

img = imread('goldhill256.bmp');

%%Convert the datatype of the image to double for calculation in matlab

id = double(img);

%%Assign the number of levels of Quantization

L = [32,64,128];

%%Looping through the different levels of quantization

for i = 1:3

%%setting the quantization size

q\_size = 256/L(i);

tk(1) = 0;

%%quantization process for Uniform quantization

for k = 2:(L(i)+1)

tk(k) = tk(k-1) + q\_size;

rk(k-1) = tk(k-1) + q\_size/2;

end

%%creating quantization levels for uniform quantization

for m = 1:256

for n = 1:256

for p = 1:L(i)

if (id(m,n) < tk(p + 1) && id(m,n) >= tk(p))

res\_img(m,n) = rk(p);

end

end

end

end

%%converting res\_image to double datatype for calculating MSE & PSNR

out\_img = double(res\_img);

t = 0;

for q = 1:256

for r = 1:256

t = t + (id(q,r) - out\_img(q,r))^2;

end

end

%%calulating MSE using formula

MSE = (1/(256^2))\*t;

%%calculating PSNR using formula

PSNR = 10\*log10((255^2)/MSE);

%%results

figure(1)

subplot(2,2,i+1)

imshow(uint8(out\_img))

title(['Quantized Image for L = ',num2str(L(i))])

xlabel(['MSE = ', num2str(MSE),' & PSNR =', num2str(PSNR)])

hold on

end

hold off

figure(1)

subplot(2,2,1)

imshow(img)

title('Orignal Image')

%%saving figure file

saveas(gca,'Uniform Quantizer','jpg')

*RESULTS:*

**

1. CONTRAST QUANTIZER

*The code has been written according to the pseudocode provided. Comments are added in the MATLAB script to explain the algorithm*

*MATLAB CODE:*

%%MATLAB implementation of Contrast Quantizer

%%Read the image

img = imread('goldhill256.bmp');

%%Convert the datatype of the image to double for calculation in matlab

id = double(img);

%%Assign the number of levels of Quantization

L = [40,60,80];

%%Looping through the different levels of quantization

for i = 1:3

%%quantization process for Contrast quantization

%%Assign the values of alpha and beta

alpha = 1;

beta = 1/3;

%%converting Luminance to contrast domain

c = alpha \* id(:,:).^beta;

%%new max and min according to c

tk(1) = min(min(c));

tk(L(i)+1) = max(max(c));

%%setting the quantization size

q\_size = (tk(L(i)+1)-tk(1))/L(i);

rk(1) = tk(1) + q\_size;

for k = 2:(L(i)+1)

tk(k) = tk(k-1) + q\_size;

rk(k-1) = tk(k-1) + q\_size/2;

end

%%creating quantization levels for contrast quantization

for m = 1:256

for n = 1:256

for p = 1:L(i)

if (c(m,n) < tk(p + 1) && c(m,n) >= tk(p))

res\_img(m,n) = rk(p);

end

end

end

end

%%Converting contrast back to Luminance

out\_img = ((res\_img).^(1/beta))/alpha;

%%converting image to double for calculating MSE and PSNR

out\_img = double(out\_img);

t = 0;

for q = 1:256

for r = 1:256

t = t + (id(q,r) - out\_img(q,r))^2;

end

end

%%calculating MSE according to formula

MSE = (1/(256^2))\*t;

%%calculating PSNR according to formula

PSNR = 10\*log10((255^2)/MSE);

%%results

figure(1)

subplot(2,2,i+1)

imshow(uint8(out\_img))

title(['Quantized Image for L = ',num2str(L(i))])

xlabel(['MSE = ', num2str(MSE),' & PSNR =', num2str(PSNR)])

hold on

end

hold off

figure(1)

subplot(2,2,1)

imshow(img)

title('Orignal Image')

%%saving figure

saveas(gca,'Contrast Quantizer','jpg')

*RESULTS:*



1. PSEUDORANDOM QUANTIZER

*The code has been written according to the pseudocode provided. Comments are added in the MATLAB script to explain the algorithm*

*MATLAB CODE:*

%%MATLAB implementation of Pseudorandom Quantizer

%%Read the image

img = imread('goldhill256.bmp');

%%Convert the datatype of the image to double for calculation in matlab

id = double(img);

%%Assign the different noise values for A

A = [22,89,113];

%%Number of bits

L = 2^3;

%%Looping through the different levels of noise

for i = 1:3

%%quantization process for Pseudorandom quantization

%%generating pseudorandom noise

ps\_ran = randi([-A(i),A(i)],256,256);

%%adding image and noise

c = id(:,:,1) + ps\_ran;

%%new min and max values according to c

tk(1) = min(min(c));

tk(L+1) = max(max(c));

%%setting the quantization size

q\_size = (tk(L+1)-tk(1))/L;

rk(1) = tk(1) + q\_size;

for k = 2:(L+1)

tk(k) = tk(k-1) + q\_size;

rk(k-1) = tk(k-1) + q\_size/2;

end

%%creating quantization levels for pseudorandom quantization

for m = 1:256

for n = 1:256

for p = 1:L

if (c(m,n) < tk(p + 1) && c(m,n) >= tk(p))

res\_img(m,n) = rk(p);

end

end

end

end

%%subtracting noise from image

res\_img = res\_img - ps\_ran;

%%converting image to double datatype for calculating MSE and PSNR

out\_img = double(res\_img);

t = 0;

for q = 1:256

for r = 1:256

t = t + (id(q,r) - out\_img(q,r))^2;

end

end

%%calculating MSE and PSNR using formula

MSE = (1/(256^2))\*t;

PSNR = 10\*log10((255^2)/MSE);

%%results

figure(1)

subplot(2,2,i+1)

imshow(uint8(out\_img))

title(['Quantized Image for L = 8 and A = ',num2str(A(i))])

xlabel(['MSE = ', num2str(MSE),' & PSNR =', num2str(PSNR)])

hold on

end

hold off

figure(1)

subplot(2,2,1)

imshow(img)

title('Orignal Image')

%%saving figure

saveas(gca,'Pseudorandom Quantizer','jpg')

*RESULTS:*



***CONCLUSION:***

* According to the results it can be shown that the MSE and PSNR values in Uniform quantization is the least and the image obtained is very similar to the original image.
* Another observation that can be seen from the contrast quantizer is that the PSNR values remain almost similar in the different levels of quantization.
* The Pseudorandom quantization shows that the output is grainy because of the added noise. As the noise increases, the image becomes grainier.