

Real-Time Digital Signal Processing
Prof. Rathna G. N.
Department of Electrical Engineering
Indian Institute of Science – Bengaluru

Lecture – 49
M3U31 - Discrete Cosine Transform - III

Namaste, welcome back to Real-Time Digital Signal Processing Course last class we discussed about Discrete Cosine Transform in 1-D. How we are going to reconstruct the 2-D form? So, today we will continue discrete cosine transform.

(Refer Slide Time: 00:38)



So, as we are continuing in the previous class, as I said from 1-D DCT, we reconstructed for 2-D DCT.

(Refer Slide Time: 00:48)

Inverse 2D-DCT

2D-DCT gives us $Y = C(XC^T)^T$ which can be rewritten

$$Y = CXC^T$$

Since C is an orthogonal we can solve for X using the fact

Therefore, $X = CTYC$, $C^{-1} = C^T$

So, today we will see how we can do? That is inverse of 2-D DCT how we are going to implement it? So, forward DCT what we had seen the thing. So, we know that Y is given as C into X transpose into a transpose what we will be taking it. So, this is X transpose what we have it here. And then the transpose of it will be giving as this way. What is it? $Y = C$ into X into C transpose what we can write it.

Since C is an orthogonal what we had seen in the last class. So, we can solve for X using the fact that the inverse of C inverse what it is going to be equal to C transpose, basically what we have assumed it. We have shown because of the orthogonal property. So, therefore, what happens to this, it will be equivalent to C transpose Y into C .

(Refer Slide Time: 01:51)

Reconstructing the Image

• In Mathematical terms:

- Let $X = (x_{ij})$ be a matrix of n^2 real numbers
- $Y = (y_{kl})$ be the 2D-DCT of X
- $a_0 = 1/\sqrt{2}$ and $a_k = 1$ for $k > 0$

• Then:

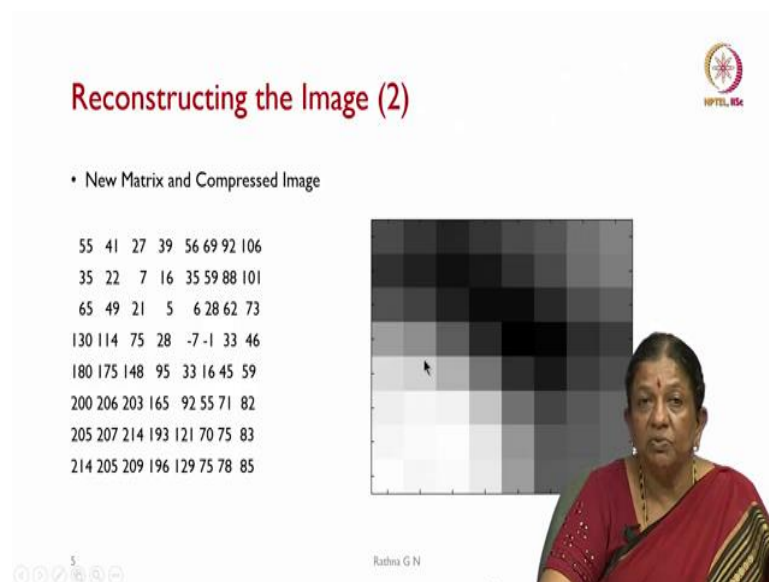
$$P_n(s, t) = \frac{2}{n} \sum_{k=0}^{n-1} \sum_{l=0}^{n-1} y_{kl} a_k a_l \cos \frac{k(2s+1)\pi}{2n} \cos \frac{l(2j+1)\pi}{2n}$$

- Satisfies $P_n(l, j) = x_{ij}$ for $j = 0, \dots, n-1$

So, how we are going to reconstruct this image in mathematical terms, let $X = x_{ij}$ be a matrix of n squared real numbers and Y will be y_{kl} be the 2-D DCT of X what will be taking it. And we know the A naught coefficient is going to be $1/\sqrt{2}$ and a_k is going to be 1 for k greater than 0 . Then what is our P_n s that is t going to be represented, as in the 2-D, 2 by n is our scaling factor k will be going from 0 to $n - 1$ that is the size of our matrix.

And because it is a 2-D, so, I will be having a double summation. So, y_{kl} into a_k into a_l . So, into $\cos \frac{k(2s+1)\pi}{2n}$ into $\cos \frac{l(2j+1)\pi}{2n}$. So, this is how our 2-D DCT is going to be represented. And we say that this satisfies, $P_n^{-1} y_{kl} = x_{ij}$ for i, j is varying from 0 to $n - 1$.

(Refer Slide Time: 03:16)



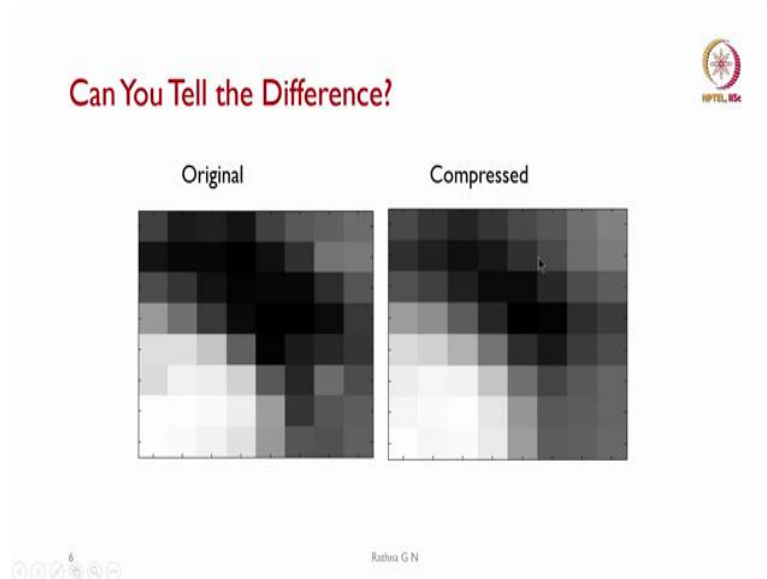
The slide is titled "Reconstructing the Image (2)" in red text. It features a logo in the top right corner. Below the title, there is a bullet point: "• New Matrix and Compressed Image". To the left of a grayscale image, there is a list of numerical values representing the new matrix. The values are arranged in 8 rows and 8 columns:

55	41	27	39	56	69	92	106
35	22	7	16	35	59	88	101
65	49	21	5	6	28	62	73
130	114	75	28	-7	-1	33	46
180	175	148	95	33	16	45	59
200	206	203	165	92	55	71	82
205	207	214	193	121	70	75	83
214	205	209	196	129	75	78	85

To the right of the matrix is a grayscale image of a person. The name "Rathna G N" is written at the bottom of the image. There are navigation icons at the bottom left of the slide.

So, how we are going to do the reconstruction of this? So, the new matrix and then what we show is the compressed image here. So, we have these are the values what it is available? And then this is the image what it has been compressed to.

(Refer Slide Time: 03:37)



So, can you make the difference between the original because I said this is a compressed image. So, in the next one we will be comparing with original and then compressed image. So, how much difference you are going to make out between the two what you can look at it. So, as you can see that here, you will be seeing the little bit of difference in the thing here. It is white, whereas here what you are seeing is as white.

And then here you have seen that all of them have white in the thing that is after compression, what we are looking at it?

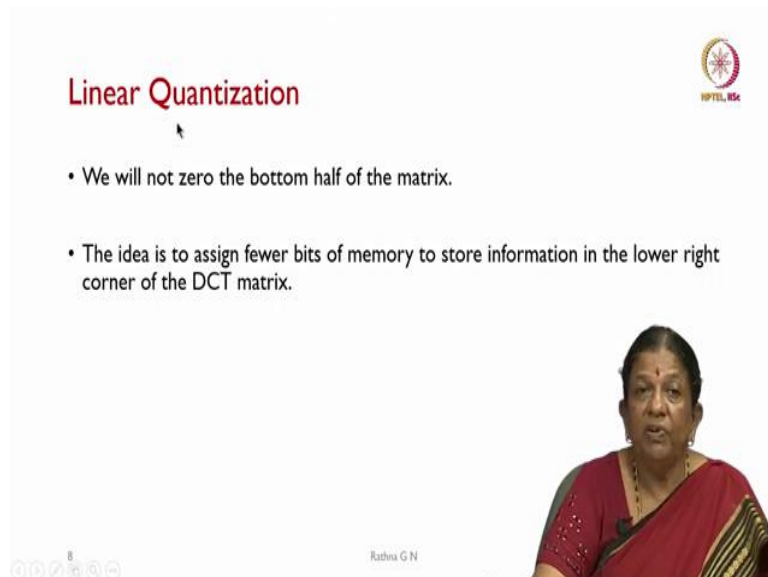
(Refer Slide Time: 04:16)



So that was this thing what is it black and white patches what you had seen the difference between the thing. Now, we will see the image basically, so, we have the original Lena image

and this is the compressed image. So, is it possible for you to find the difference minor difference? So, if you are too good enough, you can find a difference between the two.

(Refer Slide Time: 04:42)



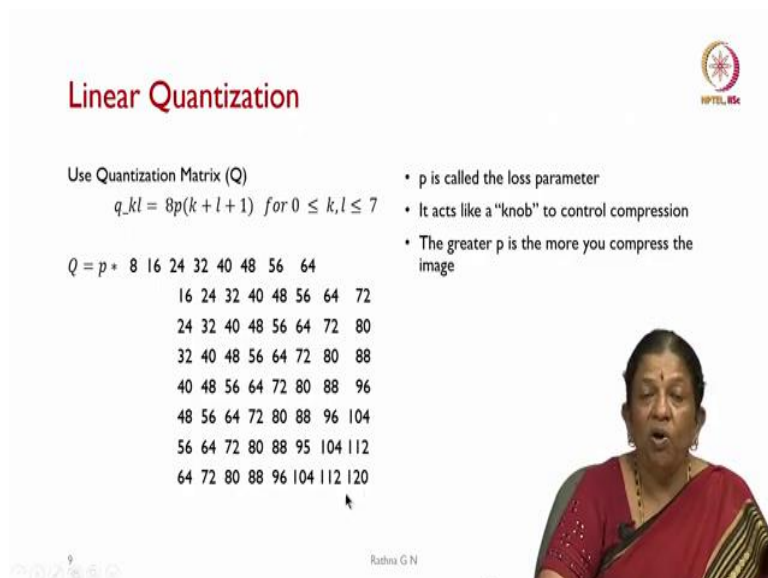
Linear Quantization

- We will not zero the bottom half of the matrix.
- The idea is to assign fewer bits of memory to store information in the lower right corner of the DCT matrix.

Rathna G N

So, what is it? We have discussed about the linear quantization in previous classes. So, we will not zero the bottom half of the matrix. So that is idea is to assign fewer bits of memory to store information in the lower right corner of the DCT matrix. Previous quantization, what we did was, A lower that is off diagonal below that what we have made everything as 0. So, here, instead of making it 0 number of bits allocated for this values is going to be made less, so that we can have the method of linear quantization included in it.

(Refer Slide Time: 05:26)



Linear Quantization

Use Quantization Matrix (Q)

$$q_{kl} = 8p(k + l + 1) \text{ for } 0 \leq k, l \leq 7$$

$Q = p *$

8	16	24	32	40	48	56	64
16	24	32	40	48	56	64	72
24	32	40	48	56	64	72	80
32	40	48	56	64	72	80	88
40	48	56	64	72	80	88	96
48	56	64	72	80	88	96	104
56	64	72	80	88	96	104	112
64	72	80	88	96	104	112	120

- p is called the loss parameter
- It acts like a "knob" to control compression
- The greater p is the more you compress the image

Rathna G N

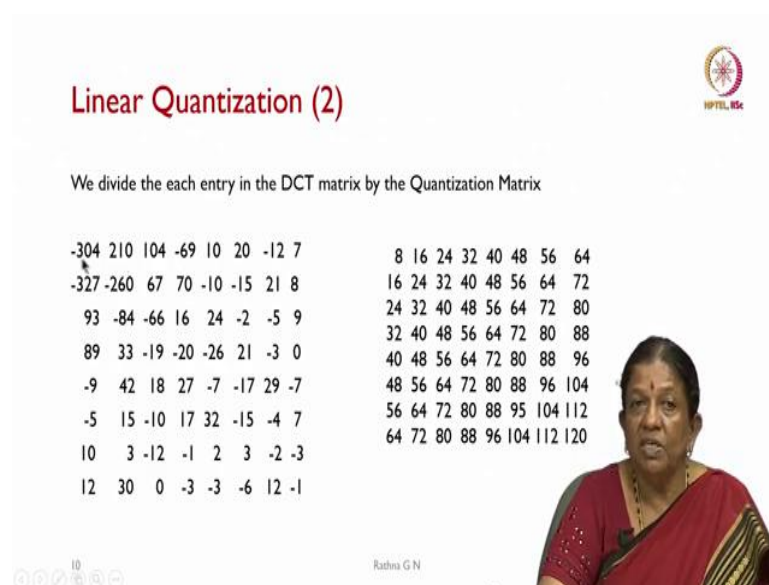
So now, how we are going to do that? So, use the quantization matrix, basically Q. So, what is it q_{kl} is $8p$ into $k + l + 1$ for $0 \leq k, l \leq 7$. So, we

say that p is called the last parameter and it is going to act like a knob to control compression. So, the greater p is the more you compress the image. So, p can vary from 1 to whatever value you are going to take it.

So, if you look at it, the quantization matrix, p multiplied with 8 that is 16, 24, 32, 40, 48, 56 and then 64 are the values. So, if you take 8 as the thing in the first row then you will be seeing that 16, 24 what you will be seeing it? This is the multiplication of what you will be seeing is 8. What you are going to assume as the Q matrix. So that means to say you will divide your DCT value with respect to this matrix.

So that whichever is low values, you will be seeing that they become very small and then you can represent them with the fewer bits.

(Refer Slide Time: 06:53)



Linear Quantization (2)

We divide each entry in the DCT matrix by the Quantization Matrix

-304	210	104	-69	10	20	-12	7	8	16	24	32	40	48	56	64
-327	-260	67	70	-10	-15	21	8	16	24	32	40	48	56	64	72
93	-84	-66	16	24	-2	-5	9	24	32	40	48	56	64	72	80
89	33	-19	-20	-26	21	-3	0	32	40	48	56	64	72	80	88
-9	42	18	27	-7	-17	29	-7	40	48	56	64	72	80	88	96
-5	15	-10	17	32	-15	-4	7	48	56	64	72	80	88	96	104
10	3	-12	-1	2	3	-2	-3	56	64	72	80	88	96	104	112
12	30	0	-3	-3	-6	12	-1	64	72	80	88	96	104	112	120

As you can see that one each entry in the DCT matrix by the quantization matrix so, this is our DCT matrix and this is our quantization matrix. So, what we are going to do is? Divide each element by each of these values. The first one is our DC which is going to be divided by 8.

(Refer Slide Time: 07:16)

Linear Quantization (3)



p = 1

```
-38 13 4 -2 0 0 0 0
-20 -11 2 2 0 0 0 0
 4 -3 -2 0 0 0 0 0
 3 1 0 0 0 0 0 0
 0 1 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
```

• New Y: 14 terms

-38
 (64)
 2^6

p = 4

```
-9 3 1 -1 0 0 0 0
-5 -3 1 0 0 0 0 0
 1 -1 0 0 0 0 0 0
 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0
```

• New Y: 10 terms

$0 - 15$
 $4b.15$

$$\frac{-304}{9} = \underline{\underline{32}}$$

$$\frac{-9 \times 32}{8}$$

So, what happens? It will be when $p = 1$ this becomes -38 . So, you can compute and then cross verify whether you are getting it correctly or not. So, the next one is a 210 is going to be divided by 16. So, you will be seeing that is the 13 value. So, you will be seeing for the rest of the values automatically lot of them have got zeros. And you will be seeing that there are few coefficients which are greater than 0 are left on this.

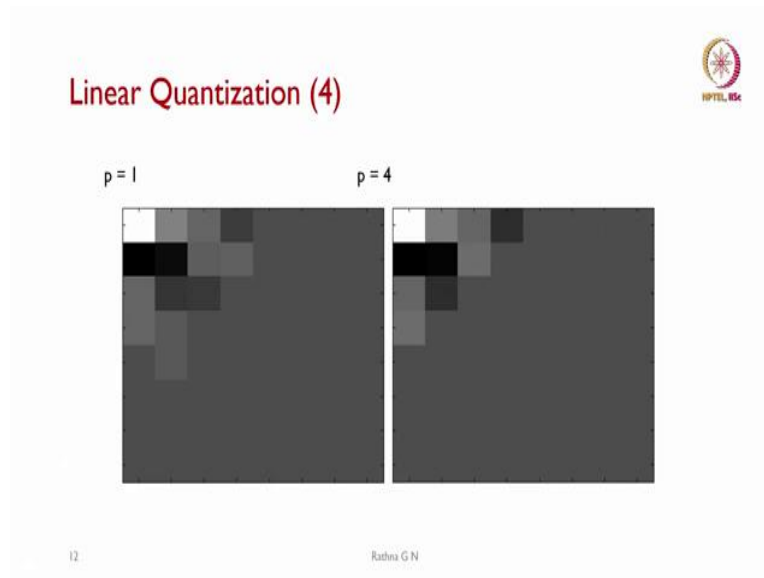
And then here what you will be seeing it is few of them in what we call it as this is the DC coefficient. These are the few AC coefficients on the top corner of what is left. So, when I use $p = 4$. So, what happens to that? Can you guess that? when $p = 1$ what we had the first one was 8 and 16. So, whereas when p is equal to that is this is $p = 1$ when $p = 4$. What will be the value of it? So, you can see that this becomes 32 and so on.

You can multiply with the thing and then you will be seeing that minus whether you are going to go it is easier to do -9 into your 32, whether it belongs to whatever value you want to get is -3 not 4. So, are we getting it or you can do -304 divided by 9 that will be the easiest way of representing it. What is the p value what I have chosen? So, approximately as I am putting it, it is 32. So, you eliminate the thing only retain the integer values.

So, you can see that compared to this, the value has got quite reduced. So, number of bits to represent this is very fewer bits. So, can you guess how many bits I need it? Because I can go 0 to 15 with 4-bits whereas -38 means I can I have to go to the power of 2 always? So, 64 is the maximum what I have to represent? So, which I will be requiring how many bits can you compare here in this case?

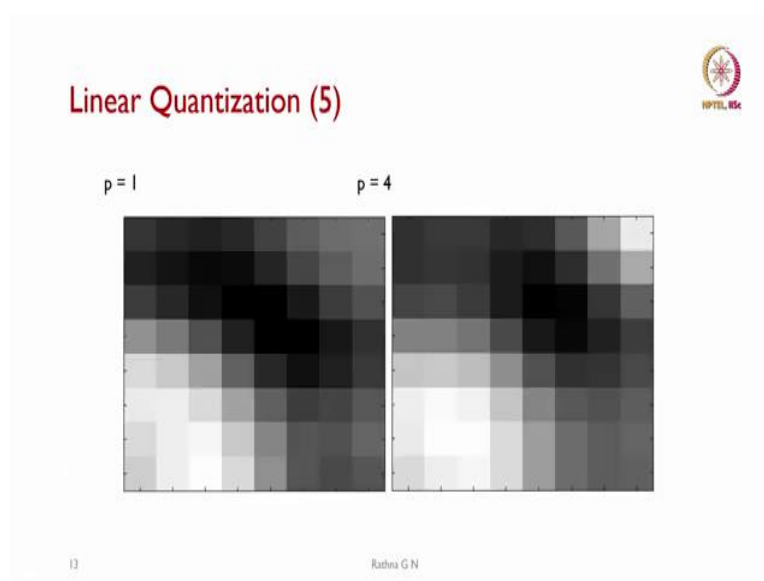
So, 22 to the power of 6 is 64 I need 6 bit. So, 6 bits to represent this value, so, I have reduced by 2 more bits in this case. So, it depends on how much detail I want to retain? So, by using $p = 4$ whether I can do with that is what it says? There are 14 terms left in Y in the case of $p = 1$ whereas in the new $p = 4$ we are left with only 10 terms. So, 4 terms have come down in the thing and then the number of bits representation for this is also reduced.

(Refer Slide Time: 10:47)




So, can you see now, after doing that $p = 1$ and then $p = 4$. So, what are the differences you are making out between the 2? That is what one has to consider and then see what kind of quantization matrix you want to use it to represent your output.

(Refer Slide Time: 11:08)



So, you will be seeing using the linear quantization with respect to $p = 1$ and then $p = 4$. So, how you are changing the thing? You can see here by grey-value it has gone to completely white and then some of it what you can find the difference.

(Refer Slide Time: 11:29)



Memory Storage

- The original image uses one byte (8 bits) for each pixel.
- Therefore, the amount of memory needed for each 8×8 block is:

$8 \times (8^2) = 512 \text{ bits}$

Linear Quantization

p	Total bits	Bits/pixel
X	512	8
1	249	3.89
2	191	2.98
3	147	2.30

14
Radhika G N

But coming to the image you mean you have not noticed $p = 1$. Later on, we will see, with $p = 4$ how much difference we are going to get it? So, in terms of memory storage, what we will look at it? So, the original image, what we had it 1 by that is 8 bits per pixel what we wanted it to be represented? So, the amount of memory needed was 8 into 8 that is a block is basically 8 into 8 square what we wanted? That is 512 bits what we want.

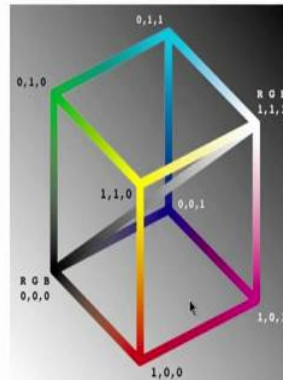
For this block 8 into 8 into 8 bits per pixel, so that is how we will be needing 512 bits. So, doing the linear quantization p if we do not use any of the value of p then we need 512 bits and the number of bits per pixel. What we need it is 8 whereas if we use $p = 1$ then total number of bits what will be having is only 249. You can go back and then check in one of the cases and then number of bits per 6 pixel what we need is 3.89.

So, like that a 2 and then 3 you will be seeing it that is total bits in $p = 2$ is 191 whereas in 3 it is 147. So, you will be seeing that if we use $p = 3$ if we are not going to lose much of the information then only 2.3 bits per 6 pixel, What we need it to represent it. So, this is how number of bits and then our memory storage is going to come down.

(Refer Slide Time: 13:03)

JPEG Imaging

- It is fairly easy to extend this application to color images.
- These are expressed in the RGB color system.
- Each pixel is assigned three integers for each color intensity.



We have seen the jpeg imaging, so, what it says? It is fairly easy to extend this application to colour images. What we have seen was the black and white in the previous case? So, you will be wondering why the thing this is for the black and white when I use a colour, as we discussed in the first class of DCT that it has to be represented RGB. So, we will be multiplying by three colours, basically.

So, you will be seeing that each pixel is assigned three integers for each colour intensity. So, what is it, here It is, you will be seeing red. You can see that R is 1 G is 0 and B is 0 in this. As you move across, so, you will be seeing that something here, dark pink what you have it which will be having both R component red component and then a blue component. And then green is going to be 0 in this.

When you come to white, you will be seeing that all of them are 1, $RGB = 1$ it becomes white. And all of them are 0s it becomes black, whereas when you have only green part of it has to be represented, you will be seeing that it is going to be 0, 1, 0 here. And the combination now from here to there you will be seeing that red is 0 here. And then you will have the green component and then blue component which will be mixing from both the ends.

And you will be seeing on the other side, also as it is represented with the cube. How different colours are represented with your RGB.

(Refer Slide Time: 14:52)

The Approach



- There are a few ways to approach the image compression.
 - Repeat the discussed process independently for each of the three colors and then reconstruct the image.
 - Baseline JPEG uses a more delicate approach.
 - Define the luminance coordinate to be:
$$Y = 0.299R + 0.587G + 0.114B$$
 - Define the color differences coordinates to be:
 - $U = B - Y$
 - $V = R - Y$



18

Rathna G N

So, how to go with this basically? A few ways to approach the image compression that is repeat the discussed process independently for each of the three colours and then reconstruct the image. So, we will be doing the compression in the R plane and then G plane and then the B plane and then we can mix it and then do it. So that is what it says? Baseline jpeg uses a more delicate approach. How it is done?

That is usually we represent with respect to luminance. What is that, coordinate to be, Y coordinate? So, what how the value is going to be defined? So, you will be taking 0.299 of R plus that is, you will be seeing 50 of the green value what you will be taking it. And then the lowest is 0.114 of the blue component. Because green is more perceptive to our eyes so which is given more weightage.

And you will be seeing that one fourth of it is given to red and then the blue gets the least preference. And then you will be defining the colour differences coordinates. So then we call it as YUV representation. U is going to have $B - Y$ and then we will be having $R - Y$. So, with luminance and then U and V component by UV what we can represent our colour images also?

(Refer Slide Time: 16:28)

More on Baseline



- This transforms the RGB color data to the YUV system which is easily reversible.



- It applies the DCT filtering independently to Y, U, and V using the quantization matrix Q_Y .

So, how it is going to be? The transformation what you will be saying RGB, what you have it? That is this is our G is here and then B you will be shifting to YUV system which is both the ways possible. So, you have been shown from here to here from YUV you can get back your RGB also that is what, what it means that it is reversible. So, you will be seeing a few of the colours that is from 3-dimensional.

What you have done is come down to only 2-dimensional. What you have is U and V what are the coordinates with your colours have been mapped into. So, it applies that DCT filtering independently to Y U and V using the quantization matrix Q_r what we can use it? And then apply it and then get the result.

(Refer Slide Time: 17:26)

JPEG Quantization



Luminance:

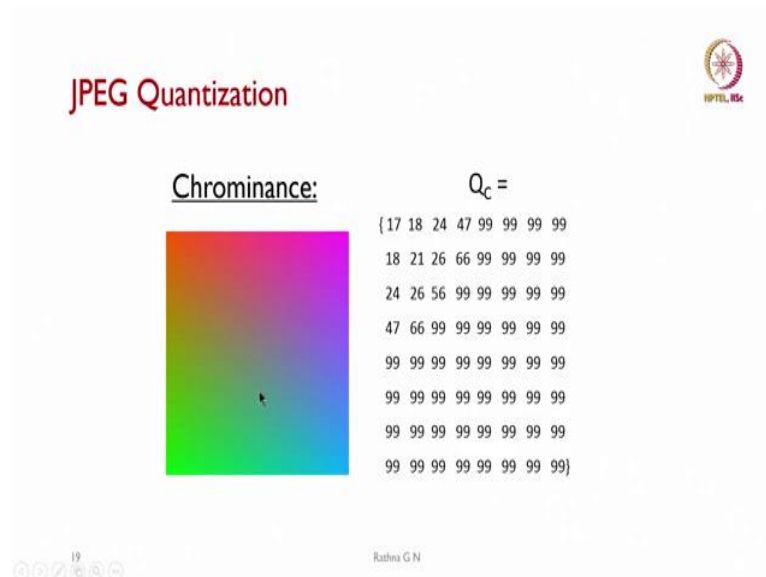


$Q_Y =$

p { 16 11 10 16 24 40 51 61
12 12 14 19 26 58 60 55
14 13 16 24 40 57 69 56
14 17 22 29 51 87 80 62
18 22 37 56 68 109 103 77
24 35 55 64 81 104 113 92
49 64 78 87 103 121 120 101
72 92 95 98 112 100 103 99 }

So, how is the luminance, so, I think the bulb shows how it is going to glow correct that is what, what we call it as luminance? So, you will be seeing the quantization Q Y here p is given these values to represent the thing. So, you are earlier we had taken the quantization matrix, basically, $p = 1$ means it is a multiple of 8 what we have taken the thing but here you can see it is with the different p what you are representing the matrix for the quantization.

(Refer Slide Time: 18:04)



And then the chrominance what you will be seeing is? Q_c is represented with this. So, you will be seeing that most of them are 99 is the value, what it will be going? So, you will be representing these colours using the prominence value of this matrix, basically.

(Refer Slide Time: 18:26)

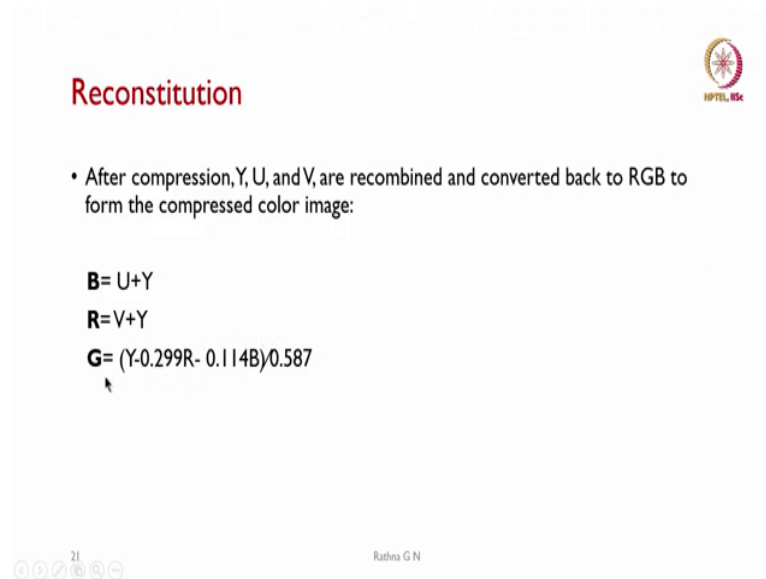
Luminance and Chrominance

- Human eye is more sensible to luminance (Y coordinate).
- It is less sensible to color changes (UV coordinates).
- Then: compress more on UV !
- Consequence: color images are more compressible than grayscale ones

So, how we are going to do that? We know that human eye is more sensible to luminance that is Y-coordinate. And it is less sensible to our colour images that are UV coordinates. So then

compress more on UV and less on Y consequences, Colour images are more compressible than gray-scale ones, as you can see it.

(Refer Slide Time: 18:51)

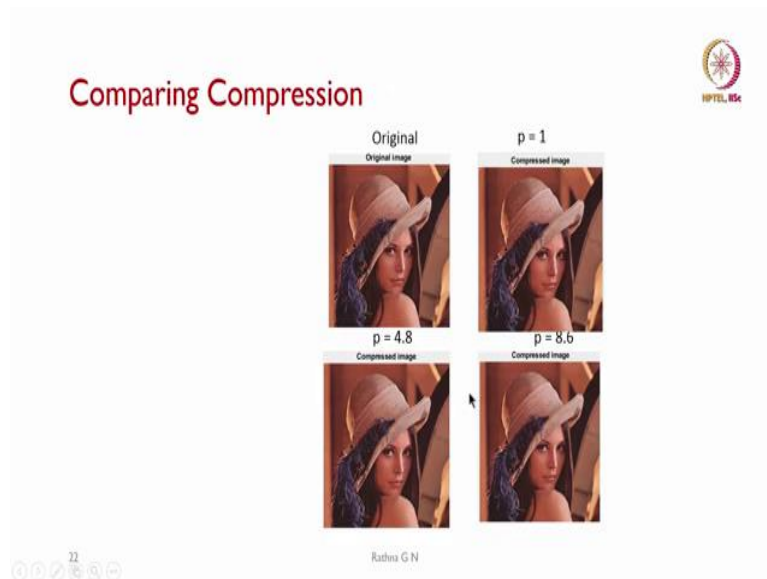


The slide is titled "Reconstitution" in red text. It features a list item: "After compression, Y, U, and V, are recombined and converted back to RGB to form the compressed color image:". Below this, three equations are listed: $B = U + Y$, $R = V + Y$, and $G = (Y - 0.299R - 0.114B) / 0.587$. The slide also includes a small logo in the top right corner and navigation icons at the bottom left.

So, how we are going to reconstitute the thing? After compression, our Y, U, V are recombined and converted back to RGB to form the compressed colour image. So, blue is represented with $U + Y$ and our red is going to be with V and then Y . And then G is going to be you are seeing that Y into 0.299 of $R - 0.114B$ that is luminance $Y - 0.299$ of red $- 0.114$ of blue whole divided by whatever we had represented our green with that is the 0.587 .

What we had represented in the original for the green we had given the weightage to calculate luminance which is divided by 0.587 . So, what is it? In the equation what you will be substituting it, basically? What I have in the equation is? This is my Y so, you will be doing $0.587 G$ equal to transform these into the other part of it. So that is why you will be getting $Y - 0.299R - 0.114B$ and to get G we have to divide this by it is weight that is 0.587 . So, this is how you will be reconstructing your green.

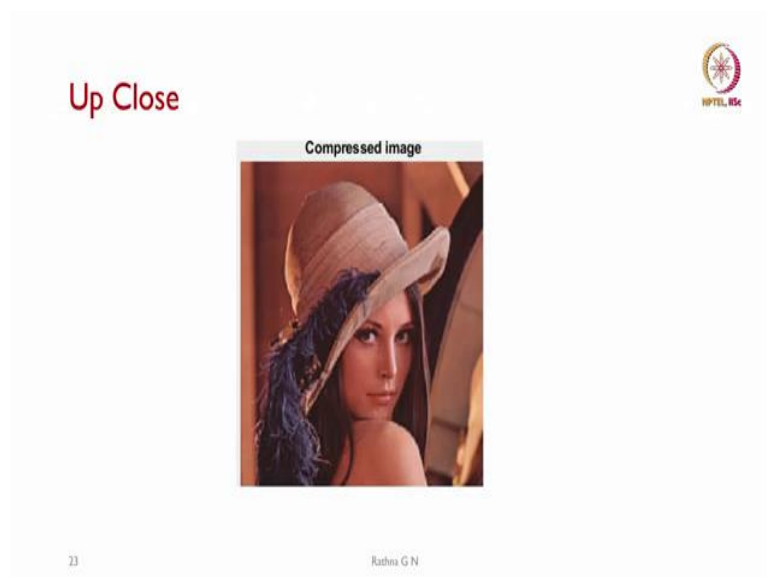
(Refer Slide Time: 20:21)



So, coming to comparing the compression, so, this is Lena has been taken. This is the original image and then, if I give $p = 1$ so, the compressed image is shown in this way. And you will be seeing the other compression $p = 4.8$ basically, what it is chosen instead of 4 and then you can see your Lena, how it is represented? And compression of $p = 8.6$ and you are seeing your Lena here.

How much difference you are going to make out? Only you can subtract from the original image and then represent that value as an image again. And then you will see how much you have lost compared to the original? But still your eyes is unable to unless you have a very sharp eyes where you can pinpoint that this is the difference, what I am seeing from the original to the compressed image.

(Refer Slide Time: 21:29)



So, this is one of the advantage of our human visual system. So, you can see compressed image we have done the blown up of it, there it was small. So, you are not deviating from much of the original. So that is what one of the compression advantage in image processing.

(Refer Slide Time: 21:50)



So, this is what we have looked at by selection of p ? How we can do the compression? And in the next class we will see it is like our fast FFT what we did for Fourier transform. So, we can do fast DCT using our butterfly structure. Here, it is not exactly butterfly but flow diagram what we will look at it in the next class. Thank you for listening to this class, we will come back in the next class for DCT. Thank you.