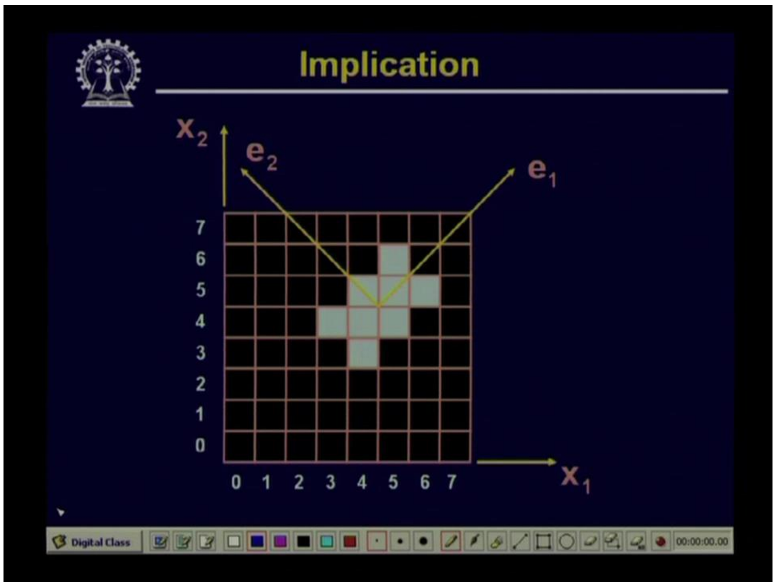
Digital Image Processing.

Professor P. K. Biswas.
Department of Electronics and Electrical Communication Engineering.
Indian Institute of Technology, Kharagpur.
Lecture-31.
K-L Transform-2.

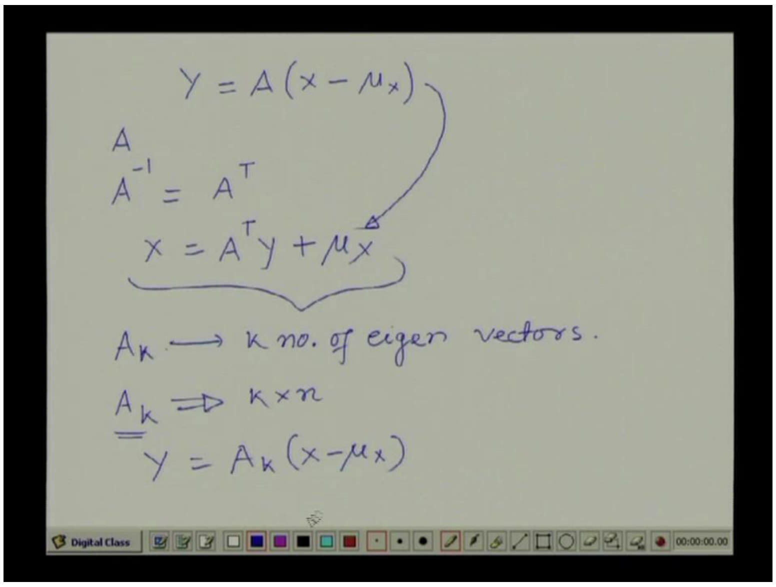
(Refer slide time: 0:38)



Implication

Now let us see the other aspects of the K-L transformation. So this is one of the applications where we have said that this this K-L transformation basically aligns the data along the vectors. Another important property of K-L transformation deals with the reconstruction of the vector x from the vector y. So by K-L transformation what we have got is, we have got a set of vectors y from another set of vectors x using the transformation matrix a where a was derived using the eigenvectors of the covariance matrix of x that is C X.

(Refer Slide Time: 1:10)



$$Y = A(x - \mu_x)$$

A

$$A^{-1} = A^T$$

$$\underbrace{x = A^T y + \mu_x}$$

$A_K \longrightarrow K$ no. of eigen vectors.

$$\underline{A_K} \Rightarrow K \times n$$

$$y = A_K(x - \mu_x)$$

So our K-L transformation expression was y equal to a into x minus mu x. Now here we find that because this matrix a, the rows of this matrix a are the eigenvectors of the covariance matrix C X. So a consists of rows which are orthogonal vectors and because rows of a are orthogonal vectors, so this simply says that inverse of a in nothing but a transpose. So now inverse of a is very simple.

If you simply take the transpose of the transform matrix a, you get the inverse of a. So from the forward transform, forward K-L transform, we can very easily find out the inverse K-L transform to reconstruct x from the transformed image or the transformed data y and in this case, the reconstruction expression is very simple. It is given by x equal to a transpose y plus mu x.

This is a direct formation from the expression of forward transformation. Now the important property of this particular expression is like this that suppose here you find that matrix A has been formed by using all the eigenvectors of the covariance matrix C X. Now suppose i choose that i will make a transformation matrix where i will not consider, i will not take all the eigenvectors of the covariance matrix C X.

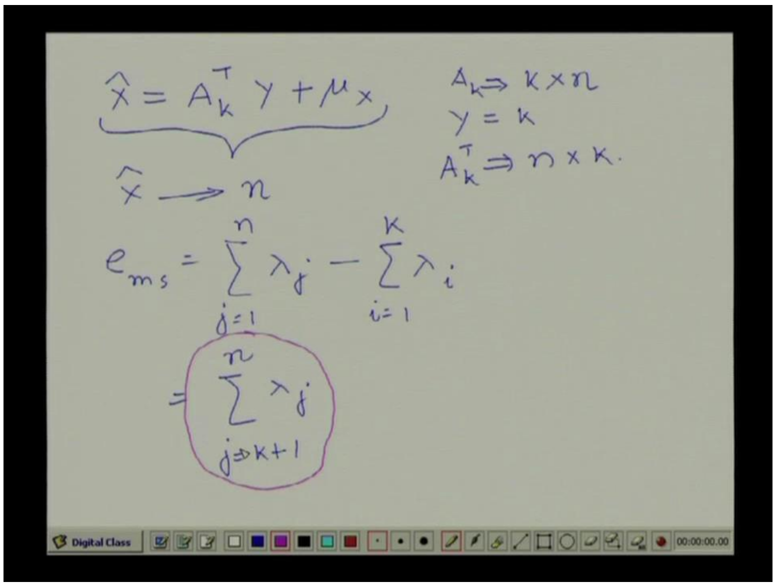
Rather i will consider say k number of eigenvectors and using that k number of eigenvectors, I will make a transformation matrix say A K. So this A K is formed using k number of eigenvectors, k number of eigenvectors of matrix C X. I am not considering all the eigenvectors of the matrix C X. And obviously because I am taking k number of eigenvectors, I will take those eigenvectors corresponding to k largest eigenvalues.

So obviously, this matrix A K now it will have k number of rows and every row will have n number of elements. So the matrix A will be of dimension k by n and the inverse transformation will be will also be done in the similar manner. So using this transformation matrix A K now i apply the transformation, so i get y equal to A K into x minus mu x. Now because A K, is of dimension k by n and x is of dimension n by 1, so naturally this transformation will generate vectors y which are of dimension k.

Now in earlier case, in our original formulation, here the transformation matrix y was the transformed vector y was of dimension n. But when i have made a reduced transformation matrix a considering only k number of eigenvectors, here i find that using the same transformation. Now the transformed vectors y that i get, they are no longer of dimension n. But this y are vectors of dimension k.

Now using these vectors of reduced dimension, if I try to reconstruct x, obviously the reconstruction will not be perfect. But what I will get is an approximate value of x. So let me write that expression like this.

(Refer Slide Time: 5:34)



$$\underbrace{\hat{x} = A_k^T y + \mu_x}$$

$$\hat{x} \longrightarrow n$$

$$A_k \Rightarrow k \times n$$

$$y = k$$

$$A_k^T \Rightarrow n \times k.$$

$$e_{ms} = \sum_{j=1}^{n} \lambda_j - \sum_{i=1}^{k} \lambda_i$$

$$= \boxed{\sum_{j=k+1}^{n} \lambda_j}$$

Here what i will get is i will get an approximate x. Sorry. I will get an approximate x. Let me write it as x hat which will be given by A K transpose y plus mu x. Now here you find, that the A K vector was of dimension k by n.

Vector y was of dimension k. Now when i take A K transpose, A K transpose becomes of dimension n by k. Now if I multiply this matrix A K transpose which of which is of dimension n by k by this vector y which is of dimension k, obviously I get a vector which is of dimension n by 1. So by this you find that this inverse transformation, it gives me the approximate reconstructed x but the dimension of x hat which is the approximation of x is same as x which is nothing but of dimension n.

So by this inverse transformation, I get back a vector x hat which is of same dimension as x but it is not the exact value of x. This is an approximate value of x and it can be shown that the mean square error of this reconstruction that is the mean square error between x and x hat is given by an expression that e m s is given by sum of lambda j where j varies from 1 to n minus sum of lambda i where i varies from 1 to k which is nothing but sum of lambda j where j varies from k plus 1 to n.

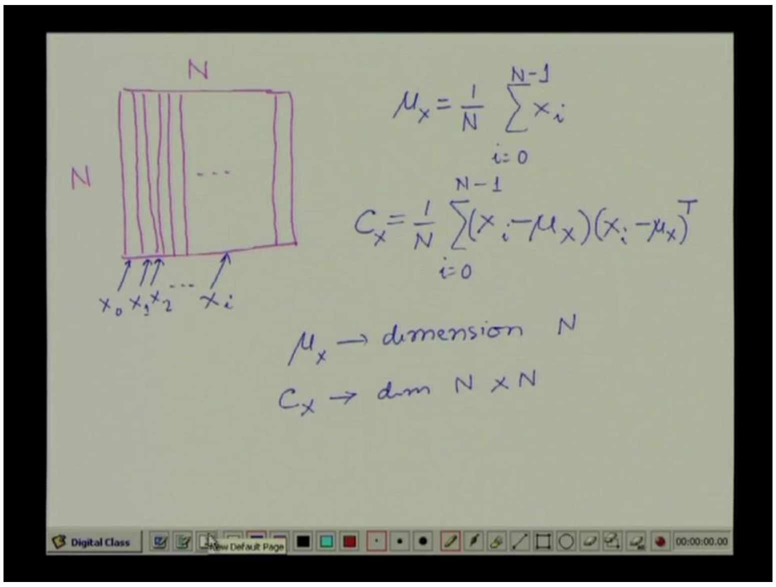
So you find that this mean square error, this term that we have got, you remember that while forming our transformation matrix A K, we have considered k number of eigenvectors of matrix C X and these k number of eigenvectors corresponding to corresponds to the largest eigenvalues of matrix C X. And in this particular expression, the mean square error is given by the sum of those eigenvalues whose corresponding eigenvectors was not considered for formation of our transformation matrix A.

And because the corresponding eigenvalues are the smallest eigenvalues. So this particular transformation and the corresponding inverse transformation ensures that the means square error of the reconstructed signal or the mean square error between x and x hat will be minimum. That is because this summation, consists of summation of only those eigenvalues which are having the minimum value.

So that is why this K-L transform is often called an optimum transform because it minimises the error of reconstruction between x and x hat. Now this is a very very important property of K-L transformation which is useful for data compression. And in this particular case, let us see that how this particular property of K-L transformation will help to reduce or to compress the image data.

So obviously the first operation you have to do is, if I want to apply this K-L transformation over an image, I have to see how to apply this K-L transformation over an image. So we have already seen a digital image is a 2 dimensional array of quantised intensity values.

(Refer Slide Time: 10:21)



N
N
$x_0\ x_1\ x_2\ \dots\ x_i$
$\mu_x = \frac{1}{N}\sum_{i=0}^{N-1} x_i$
$C_x = \frac{1}{N}\sum_{i=0}^{N-1}(x_i-\mu_x)(x_i-\mu_x)^T$
$\mu_x \rightarrow$ dimension N
$C_x \rightarrow$ dim $N \times N$
Digital Class

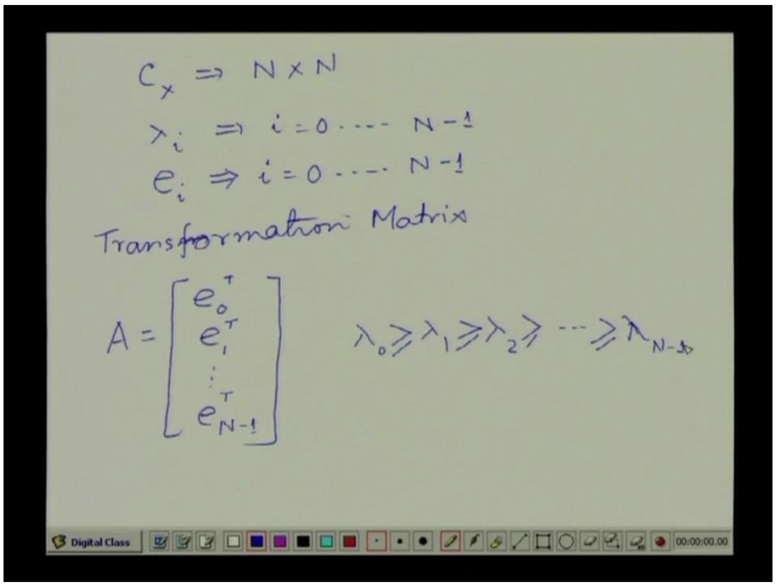
So a digital image as it is represented by a 2 dimensional array of quantised intensity values. So let us put a digital image in this form. Now here, let us assume that this image consists of n number of rows and n number of columns. So there will be n number of columns and n number of rows. And as we have said, that in order to be able to apply K-L transformation the data has to be represented by a collection of the vectors.

So this 2 dimensional image or 2 dimensional array, which consists of n number of rows and n number of columns can be converted into a set of vectors in more than one ways. So let us assume in this particular case, that we represent every column of this 2 dimensional array as a vector. So if we to do that, then every column of this, so this will be represented by a vector say x 0 this column will be represented by a vector say x 1.

This column will be represented by a vector say x 2 and this way we will have say n number of vectors as there are n number of columns. So once we have these n number of vectors for these n number of vectors, we can find out the mean vector which is mu x and this is given by 1 upon capital n then summation x i where i varies from 0 to capital n minus 1.

And similarly we can also find out the covariance matrix of these n vectors and the expression for the covariance matrix as we have already seen that this is 1 upon capital n, summation x i minus minus mu x into x i minus mu x transpose where this i will vary from 0 to capital n minus 1. And here we find that our mean vector mu x, this is of dimension capital N whereas the covariance matrix C X, this is of dimension capital N by capital N.

(Refer Slide Time: 14:00)



$C_x \Rightarrow N \times N$

$\lambda_i \Rightarrow i = 0 \cdots N-1$

$e_i \Rightarrow i = 0 \cdots N-1$

Transformation Matrix

$$A = \begin{bmatrix} e_0^T \\ e_1^T \\ \vdots \\ e_{N-1}^T \end{bmatrix} \qquad \lambda_0 \geq \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{N-1}$$

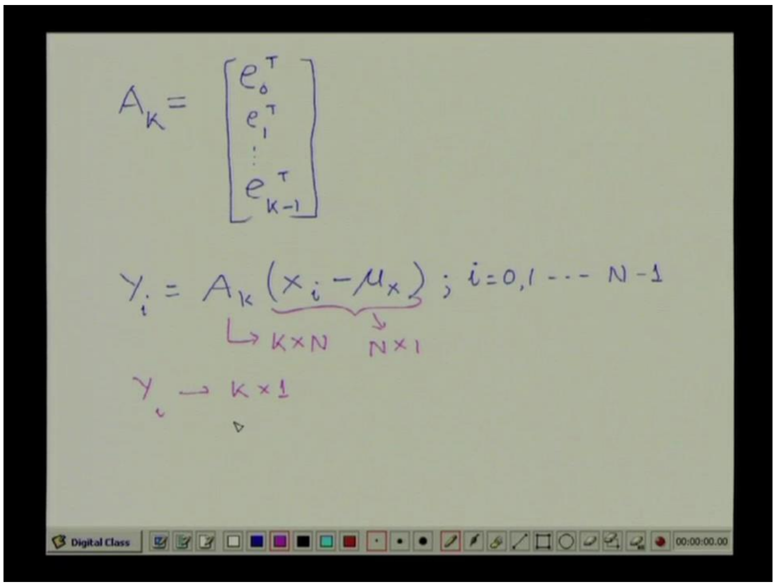
So once we have obtained the mu mean vector mu x and the covariance matrix C X, we can find out the eigenvectors and eigenvalues of this covariance matrix C X. And as we have already seen, that because this particular covariance matrix C X is of dimension capital N by capital N, there will be N number of eigenvalues lambda i where this i varies from 0 to capital n minus 1 and corresponding to every eigenvalue lambda i there will be an eigenvector e i.

So this e i, eigenvector e i, i here again will vary from 0 to capital n minus 1. So given this n number of eigenvectors for n number of eigenvalues, we can make the transformation matrix, we can form the transformation matrix A and here this transformation matrix A will be formed as say e 0 transpose. I will write this as transpose because e 0 being eigenvector and normally a vector is represented as a column vector.

So we will write the matrix A, which consists of a number of where rows of this matrix A will be the eigenvectors of the covariance matrix A covariance matrix C X. So this A will be e 0 transpose e 1 transpose and there are n number of eigenvectors so i will have e n minus 1 transpose where this e 0 corresponds to the eigenvalue lambda 0.

And obviously in this case, as we have already said that our assumption is lambda 0 is greater than or equal to lambda 1 which is greater than or equal to lambda 2 and continued like this it is greater than or equal to lambda n minus 1. So this is how we form the transformation matrix A.

(Refer Slide Time: 16:31)



$$A_K = \begin{bmatrix} e_0^T \\ e_1^T \\ \vdots \\ e_{K-1}^T \end{bmatrix}$$

$$y_i = \underset{\hookrightarrow K \times N}{A_K} \underbrace{(x_i - \mu_x)}_{N \times 1} \; ; \; i = 0, 1 \cdots N-1$$

$$y_i \rightarrow K \times 1$$

Now from this transformation matrix, we can make a truncated transformation matrix where instead of using all the eigenvectors of the covariance matrix C X, we consider only the first k number of eigenvectors which corresponds to k number of eigenvalues, k number of the largest eigenvalues. So we form the transformation matrix, the modified transformation matrix A K using the first k number of eigenvectors.

So in our case, A K will be e 0 transpose, e 1 transpose and likewise it will go up to e k minus 1 transpose. And using this A K, we take the transformation of the different column vectors of the image which we have represented by vector x i. So for every x i, we get a transform vector say y i. So here, the transformation equation is y i is equal to A K. This is the modified transformation matrix into x i minus mu x where this i varies from 0 to capital n minus 1.

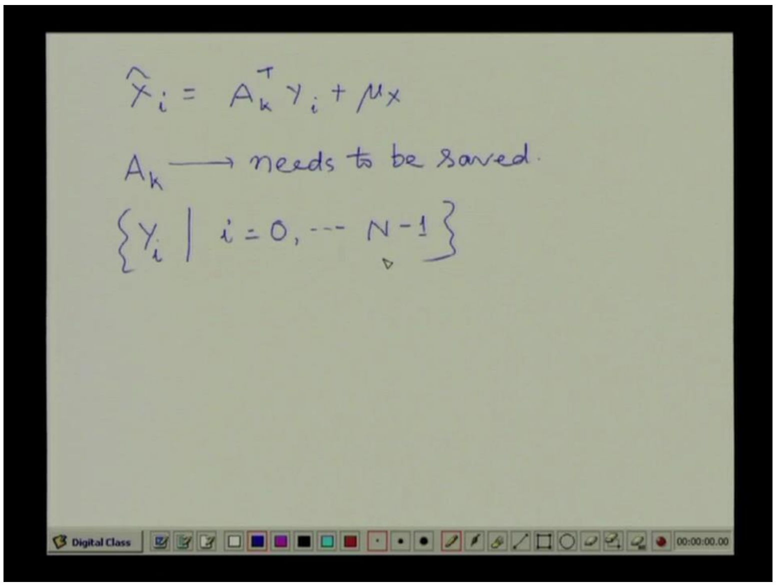
So here we find, that because A K is of dimension, the dimension of A K is k by n and dimension of x i and mu x, both of them are of dimension n by 1. So x i minus mu x, this is a vector of dimension capital n by 1. So this particular vector, this is of dimension capital n by 1. So we find that when i multiply, this when i perform this transformation A K into x i minus mu x, this actually leads to a transformed vector y i where y i will be of dimension k by 1.

So this is the dimensionality of y i. That means using this transformation with the transformation matrix A K we are getting the transform vector y i of dimension k. So if this is done, if this transformation is carried out for all the column vectors of matrix of the 2 dimensional image, in that case I get n number of transformed vectors y i where each of this transformed vector is a vector of dimension k.

That means the transformed image that I will get, the transformed image will consists of n number of column vectors where every column is of dimension k. That means the transformed image now will be of dimension k by n having k number of rows and n number of columns. You remember that our original image was of dimension capital N by capital N.

Now using this transformed image, if I do the inverse transformation, to get back the original image, as we said earlier that we do not get the perfect perfectly reconstructed image rather what we will get is an approximate image.

(Refer Slide Time: 20:12)



$\hat{x}_i = A_k^T y_i + \mu_x$

$A_k \longrightarrow$ needs to be saved.

$\{y_i \mid i = 0, \cdots N-1\}$

So this approximate image will be given by x i hat is equal to A K transpose, y i plus mu x where this x i hat, here you find that it will be of dimension capital n.

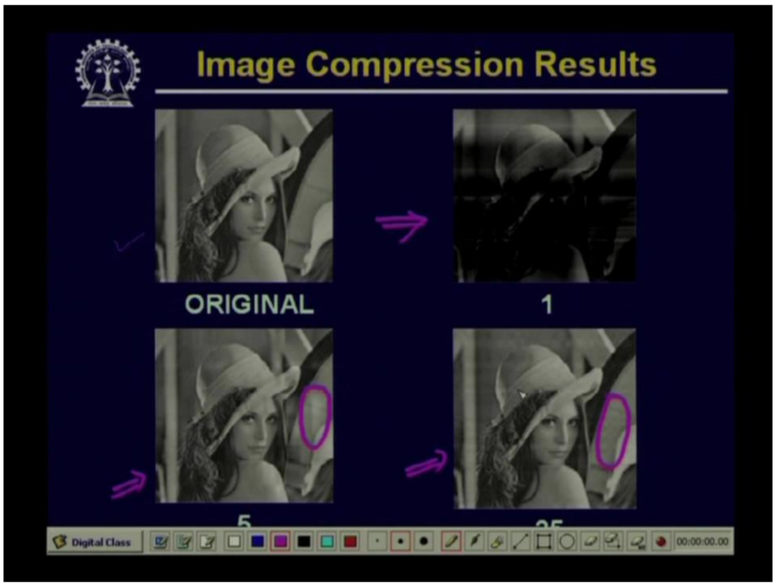
So collection of all these x i hats that gives you the reconstructed image from the transformed image. As we have said that the mean square error between the reconstructed image and the original image in this particular case will be minimum because that is how we have formed the transformation matrix and there we have said that the mean square error of the reconstructed vector from the original vector was summation of the eigenvalues which are left out, corresponding to which the eigenvectors were not considered for formation of the transformation matrix.

So here we find, that because now, in this case for getting the reconstructed image, what are the quantities that we have to save. Obviously, the first quantity that we have to save, the first information that we have to save is the transformation matrix A K. So this A K needs to be saved. And the other information that we have to save is the transformed matrix or the set of transformed vectors y i for i equal to

So the set of transformed vectors y i for i equal to 0 to capital N minus 1. So if we save these two quantities A K and the set of transformed vectors y i, then from these two quantities, we can reconstruct an approximate original image given by the vectors x i hat. So we find that in this case, the amount of compression that can be obtained depends upon, what is the value of k, that is how many eigenvectors we really consider we really take into account for formation of our transformation matrix A.

So the value of k can be 1 where we consider only one eigenvector to form our transformation matrix A. It can be two where we consider only two eigenvectors to form the transformation matrix A and depending upon the number of the eigenvectors, the amount of compression that you that we can achieve will be varying. Now let us see that what are the kind of results, that we get with different values of k.

(Refer Slide Time: 23:31)



So here you find that we have shown some of the images here the top image, this is the original image. Now when this original image is actually transformed and reconstructed using the transformation matrix with only one eigenvector. So this is the eigenvector having which corresponds to the corresponds to the largest eigenvalue of the covariance matrix.

Then the then the reconstructed image that we get is given by this result. So here you find , that the reconstructed image is not at all good but still from the reconstructed image we can make out that what this image is about. Now if we increase the number of eigenvectors in the transformation matrix. When I use 5 eigenvectors as the transformation matrix then the reconstructed image is given by this one.

You find that the amount of information which is contained in this particular image is quite improved though this is not identical with the original image. If we increase the number of eigenvectors further that is we use 25 eigenvectors to form the transformation matrix then this is the reconstructed image that we get.

Now if you closely observe between these two images, compare these two images, you find that these are some artefacts say for example in this particular region. There is an artefact something like a vertical line which was not present in the original image and that is improved to a larger extent in this particular image. So again the image quality has been improved if I increase the number of eigenvectors from 5 to 35.

Similarly if I increase the number of eigenvectors further, if I go for 50 eigenvectors, then image is further improved. 100 eigenvectors, I get further improvement. If I use 128 number of eigenvectors, I get still a better reconstructed image. So this way, you will find that if I consider all the eigenvectors of the covariance matrix to form uh the transformation matrix, in that case the reconstruction will be a perfect reconstruction.

So here, we have discussed about the K-L transformation where we have said the K-L transformation is fundamentally definED from the other transformation that we have discussed so earlier that is the discrete fourier transformation, discrete cosine transformation and so on. And there we have said that in those case of transformations, the transformation matrix or the transformation kernel is fixed.

Whereas in case of K-L transformation, you find that the transformation kernel that is the transformation matrix A which is derived from the covariance matrix and this covariance matrix actually represents what is the statistical property of the vector representation of the data. So here the kernel of transformation or the transformation matrix is dependant upon the data. It is not fixed.

So that is the fundamental difference between the other transformations with the with the K-L transformation. But the advantage of the K-L transformation that is quite obvious from the reconstructed images is that that the energy compaction property which we have said earlier, here in this particular case, in case of K-L transformation. The energy compaction property is much higher than that of any other transformation.

Here you find that the earlier result that we have shown where using only one eigenvector as the transformation matrix, this particular result here using only one eigenvector, still I can reconstruct the image and I can say what is the content of that image, though the reconstruction quality is very poor. So that shows that the energy compaction in eigenvector in the number of components is much higher in case of K-L transform than in case of other transformation.

But as is as it is quite obvious, that the computational complexity for K-L transformation is quite high compared to the other transformations and in fact that is the reason that despite strong property of energy compaction K-L transformation has not been much popular for data compression operations. Thank you.