

Digital Image Processing

Prof. P. K. Biswas

Department of Electronics and Electrical Communication Engineering

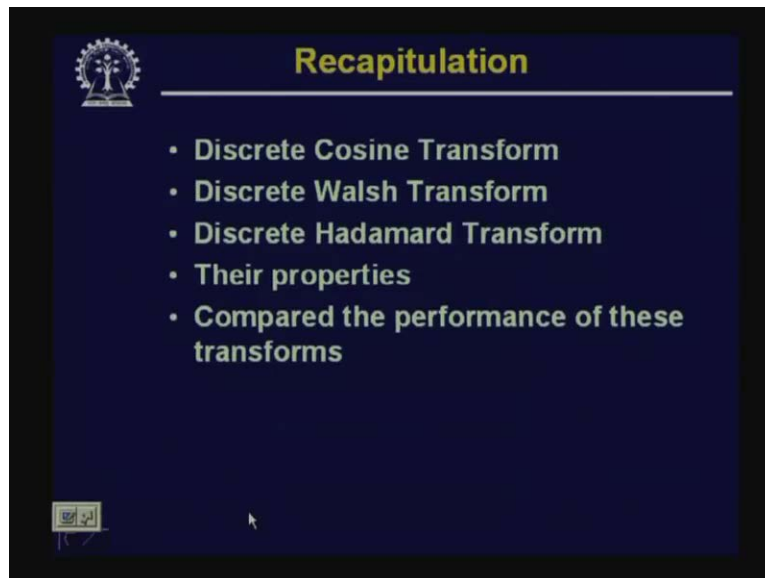
Indian Institute of Technology, Kharagpur

Lecture - 16

K – L Transform

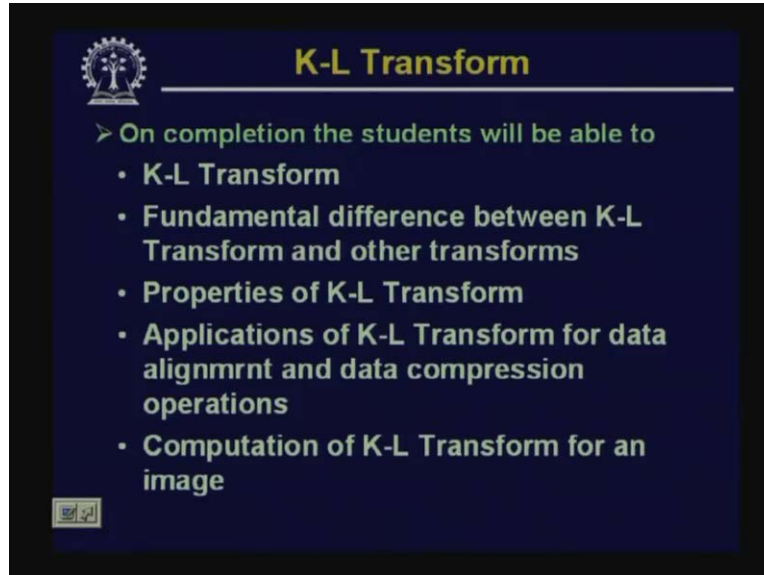
Welcome to the video lecture on digital image processing. For last few classes, we were discussing about the image transformations.

(Refer Slide Time: 1:15)



So, we have talked about the unitary transformation, we have talked about the Fourier transformation and in the last class, we have talked about the discrete cosine transform, we have seen the discrete Walsh transform, discrete Hadamard transform. We have seen their properties and we have compared the performance of these transformation operations.

(Refer Slide Time: 1:32)



The slide features a dark blue background with a white tree logo in the top left corner. The title 'K-L Transform' is written in yellow at the top center. Below the title, a green arrow points to the text 'On completion the students will be able to'. A list of five bullet points follows, all in white text. A small logo is visible in the bottom left corner of the slide.

K-L Transform

- On completion the students will be able to
 - K-L Transform
 - Fundamental difference between K-L Transform and other transforms
 - Properties of K-L Transform
 - Applications of K-L Transform for data alignment and data compression operations
 - Computation of K-L Transform for an image

In today's lecture, we will talk about another transform operation which is fundamentally different from the transformations that we have discussed in last few classes. So, the transformation that we will discuss about today is called K – L transformation. We will see what is the fundamental difference between K – L transform and other transformations, we will see the properties of K – L transform, we will see the applications of K – L transform for data alignment and data compression operations and we will also see the computation of K – L transform for an image.

Now, as we said that K – L transform is fundamentally different from other transformations; so before we start discussion on K – L transform, let us see what is the difference. The basic difference in all the previous transformations that we have discussed that is whether it is the Fourier transformation or discrete cosine transformation or Walsh transformation or Hadamard transformation; in all these cases, the transformation kernel whether it is forward transformation kernel or inverse transformation kernel, they are fixed.

(Refer Slide Time: 2:51)

$$\text{DFT}$$
$$g(x, u) = e^{-j \frac{2\pi}{N} ux}$$
$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$
$$\mu_x = E\{X\}$$
$$C_x = E\{(x - \mu_x)(x - \mu_x)^T\}$$
$$\rightarrow n \times n$$

So, for example, in case of discrete Fourier transformation or DFT, we have seen that the transformation kernel is given by $g(x, u)$ is equal to $e^{-j \frac{2\pi}{N} ux}$. Similarly, for the discrete cosine transformation as well as for other transformations like Walsh transform or Hadamard transform. In all those cases, the transformation kernels are fixed. The values of the transformation kernel depend upon the locations x and the location u . The kernels are independent of the data over which the transformation has to be performed.

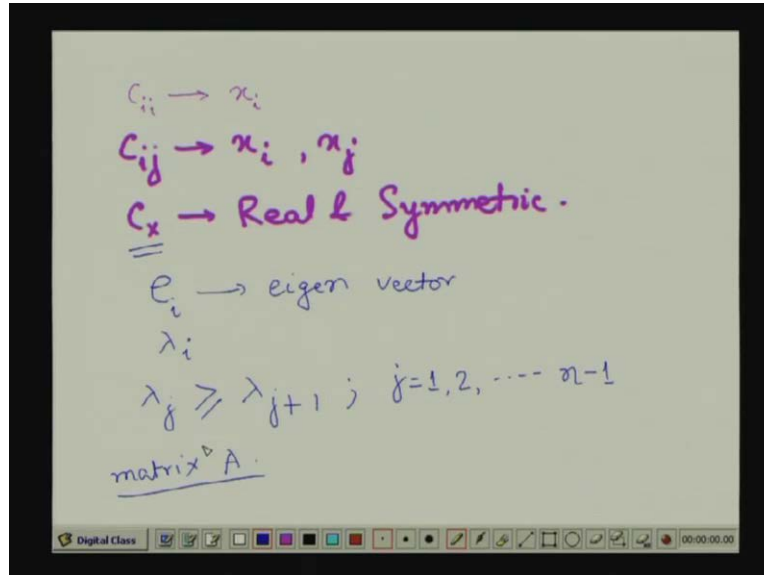
But unlike these transformations, in case of $K - L$ transformation, the transformation kernel is actually derived from the data. So, in case of $K - L$ transform, it actually operates on the basis of statistical properties of vectored representation of the data.

So, let us see how these transformations are actually opted. So, to go for $K - L$ transformation, our requirement is the data has to be represented in the form of vectors. So, let us assume a population of vectors say x which are given like this. So, we consider a vector population x which is given by say x_1, x_2, x_3 say upto x_n . So, these vectors x are actually vectors of dimension n .

Now, given such a state of vectors or population of vectors x , we can find out the mean vector given by μ_x which is nothing but the expectation value of this vector population x and similarly, we can also find out the covariance matrix C_x which is given by the expectation value of x minus the mean vector μ_x into x minus μ_x transpose.

So here, you will find that x , since x is of dimension n ; this particular covariance matrix will be of dimension n by n . So, this is the dimensionality of the covariance matrix C_x and obviously, the dimensionality of the mean vector μ_x will be equal to n .

(Refer Slide Time: 5:57)



Now, in this covariance matrix C_x , you will find that an element C_{ii} that is an element in the i 'th row and i 'th column is nothing but the variance of the element x_i of the vectors x . Similarly, an element C_{ij} , this is nothing but the covariance of the elements x_i and x_j of the vectors x and you will find that this particular covariance matrix C_x , it is real and symmetric. So, because this covariance matrix is real and symmetric, we can always find a set of n orthonormal Eigen vectors. So, because this covariance matrix C_x is real and symmetric, we can find out a set of orthonormal Eigen vectors of this covariance matrix C_x .

Now, if we assume that suppose e_i is an Eigen vector of this covariance matrix C_x which corresponds to the Eigen value λ_1 λ_i . So, corresponding to the Eigen value λ_i , we have the Eigen vector say e_i and we assume this Eigen values are arranged in descending order of magnitude of the Eigen values. That is we assume that λ_j is greater than or equal to λ_{j+1} for j varying from 1, 2 upto n minus 1.

So, what we are taking? We are taking the Eigen values of the covariance matrix C_x and we are taking the Eigen vectors corresponding to every Eigen value. So, corresponding to the Eigen value λ_i , we have this Eigen vector e_i and we also assume that these Eigen values are arranged in descending order of magnitude that is λ_j is greater than or equal to λ_{j+1} for j varying from 1 to n minus 1.

Now, from this set of Eigen vectors, we form a matrix, say A . So, we form matrix A from this set of Eigen vectors and this matrix A is formed in such a way that the first row of matrix A is the Eigen vector corresponding to the largest Eigen value and similarly the last row of this matrix A corresponds to the Eigen vector is the Eigen vector which corresponds to the smallest Eigen value of the covariance matrix C_x .

(Refer Slide Time: 9:50)

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

Properties

$$\mu_y = 0$$
$$C_y = A C_x A^T$$
$$C_y = \begin{bmatrix} \lambda_1 & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & 0 & \dots & 0 \\ 0 & \dots & \dots & \dots & \dots \\ \vdots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \lambda_n \end{bmatrix}$$

Now, if we use such a matrix A to obtain the transform operations, then what we get is we get a transformation of the form say y equal to A into x minus μ_x . So, using this matrix A which has been so formed, we form our transformation like y equal to A into x minus μ_x where you find that x is a vector and μ_x is the mean vector.

Now, this particular transformation, the transform output y that you get, that follows certain important relationship. The first relationship, the important property is that the mean of these vectors y or μ_y is equal to 0. So, these are the properties of the vector y that is obtained. So, the first property is **the mean of y** mean of vectors, y μ_y equal to 0.

Similarly, the covariance matrix of y given by C_y , this is also obtained from C_x , the covariance matrix of x and the transformation matrix that we have generated A . And the relationship between the covariance matrixes of y is like this that C_y is given by $A C_x A$ transpose. Not only that, this covariance matrix C_y is a diagonal matrix whose elements along the main diagonal are the Eigen values of C_x .

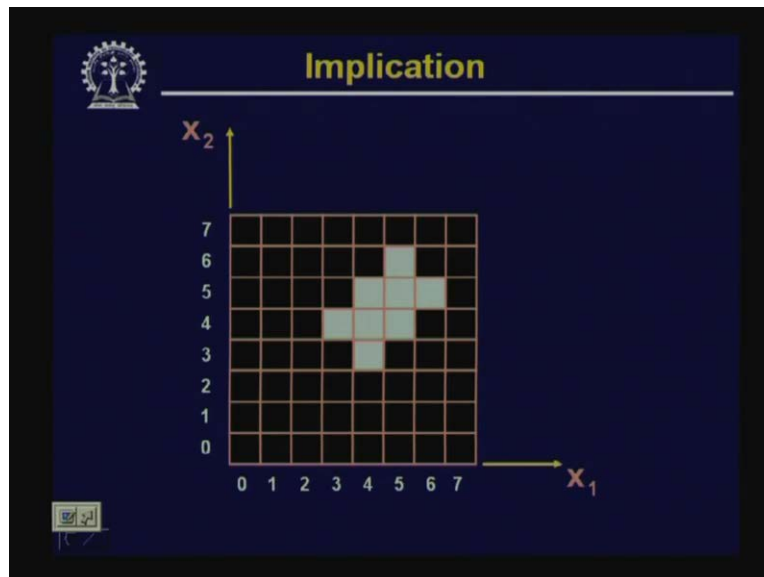
So, this C_y will be of the form λ_1 0, 0, so it continues like this; 0, then 0, λ_2 , 0, it continues like this then finally we have 0, 0, 0 and upto this we have λ_1 . So, this is the covariance matrix of y that is C_y .

And obviously, in this particular case, you will find that the Eigen values of C_y is same as the Eigen values of C_x which is nothing but λ_1 , λ_2 upto λ_n and it is also a fact that the Eigen vectors of C_y will also be same as the **Eigen n** Eigen vector of C_x and since in this case we find that the diagonal elements are always 0, that means the elements of y vectors they are uncorrelated.

So, the property of the vectors y that we have got is the mean of the vectors equal to 0. We can obtain the covariance matrix C_y from the covariance matrix C_x and the transformation matrices

A. The Eigen values of C_y are same as the Eigen values of C_x and also as the off diagonal elements of C_y are equal to 0; that indicates that the elements of the vectors y , different elements of the vector y are uncorrelated. Now, let us see what is the implication of this. To see the implication of these observations, let us come to the following figure.

(Refer Slide Time: 13:40)



So, in this figure we have a binary image, a 2 dimensional binary image. Here we assume that all the pixel locations which are white, there an object is present, an object element is present and wherever the pixel value is 0, there is no object element present.

So, in this particular case, the object region consists of the pixels say (3, 4) (4, 3) (4, 4) then (4, 5) then (5, 4) then (5, 5) then (5, 6) and (6, 5). So, these are the pixel location which contains the objects and other pixel location does not contain the object.

Now, what we plan to do is we will find out the K – L transform of those pixel locations where an object is present. So, from this, we have the population of Eigen vectors which is given by this.

(Refer Slide Time: 14:50)

$$x = \left\{ \begin{pmatrix} 3 \\ 4 \end{pmatrix}, \begin{pmatrix} 4 \\ 3 \end{pmatrix}, \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \begin{pmatrix} 5 \\ 4 \end{pmatrix}, \begin{pmatrix} 5 \\ 5 \end{pmatrix}, \begin{pmatrix} 5 \\ 6 \end{pmatrix}, \begin{pmatrix} 6 \\ 5 \end{pmatrix} \right\}$$
$$\mu_x = \begin{pmatrix} 4.5 \\ 4.5 \end{pmatrix}$$
$$C_x = E \left\{ (x - \mu_x)(x - \mu_x)^T \right\}$$

Just, we consider the locations of the pixels where an object is present that is the pixel is equal to white and those locations are considered as vectors and so the population of vectors x is given by we have (3, 4) because in location (3, 4) we have an object present. We have (4, 3), there also an object is present; we have (4, 4), here also an object is present; we have (4, 5), we have (5, 4), then (5, 5), then (5, 6) and then (6, 5).

So, we have 1, 2, 3, 4, 5, 6, 7, 8 vectors, 8 2 - dimensional vectors in this particular population. Now, from these vectors, it is quite easy to compute the mean vector μ_x and you can easily compute that mean vector μ_x in this particular case will be nothing but 4.5, 4.5. So, this is the mean vector that we have got.

So once we have the mean vector, now we can go for computing the covariance matrix and you will find that the covariance matrix C_x was defined as the expectation value of x minus μ_x into x minus μ_x transpose. So, finding out x minus μ_x into x minus μ_x transpose for all the vectors x and taking the average of them gives us the expectation value of x minus μ_x into x minus μ_x transpose which is nothing but the covariance matrix C_x .

(Refer Slide Time: 17:06)

$$x_1 - \mu_x = \begin{pmatrix} -1.5 \\ -0.5 \end{pmatrix} \Rightarrow \{ (x_1 - \mu_x) (x_1 - \mu_x)^T \} = \begin{pmatrix} 2.25 & 0.75 \\ 0.75 & 0.25 \end{pmatrix}$$
$$x_2 - \mu_x = \begin{pmatrix} -0.5 \\ -1.5 \end{pmatrix} = \begin{pmatrix} 0.25 & 0.75 \\ 0.75 & 2.25 \end{pmatrix}$$
$$x_3 - \mu_x = \begin{pmatrix} -0.5 \\ -0.5 \end{pmatrix} = \begin{pmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{pmatrix}$$
$$x_4 - \mu_x = \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 0.25 & -0.25 \\ -0.25 & 0.25 \end{pmatrix}$$
$$x_5 - \mu_x = \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} = \begin{pmatrix} 0.25 & -0.25 \\ -0.25 & 0.25 \end{pmatrix}$$
$$x_6 - \mu_x = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{pmatrix}$$
$$x_7 - \mu_x = \begin{pmatrix} 0.5 \\ 1.5 \end{pmatrix} = \begin{pmatrix} 0.25 & 0.75 \\ 0.75 & 2.25 \end{pmatrix}$$
$$x_8 - \mu_x = \begin{pmatrix} 1.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 2.25 & 0.75 \\ 0.75 & 0.25 \end{pmatrix}$$

So here, for the first vector x_1 , we can find out x_1 minus μ_x as; you find that x_1 is nothing but that the vector (3, 4), so x_1 minus μ_x will be equal to minus 1.5 and minus 0.5. So, you can find out x_1 minus μ_x into x_1 minus μ_x transpose, if we compute this this will be a value equal to 0.25, 0.75, 0.75 and 2.25.

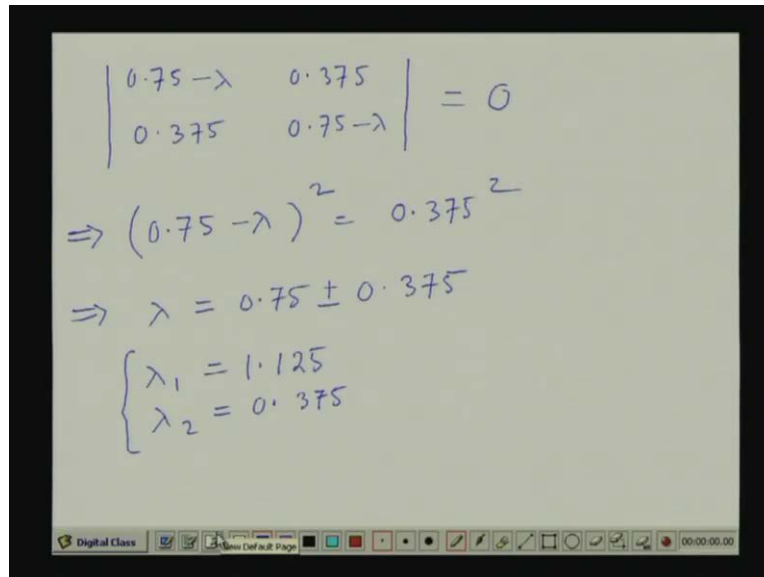
So similarly, we find out x minus μ_x into x minus μ_x transpose for all other vectors in the population x and finally, average of all of them gives us the covariance matrix C_x and if you compute like this, you can easily obtain that covariance matrix C_x will come out to be 0.75, 0.375, 0.375 and 0.75.

(Refer Slide Time: 18:15)

$$C_x = \begin{pmatrix} 0.75 & 0.375 \\ 0.375 & 0.75 \end{pmatrix}$$

So, this is the covariance matrix of the population of vectors x . Now, once we have this covariance matrix; to find out the $K - L$ transformation, we have to find out what are the Eigen values of this covariance matrix and to determine the Eigen values of the covariance matrix, you all might be knowing that the operation is like this that given the covariance matrix, we simply perform 0.75 minus λ 0.375 , then 0.375 , 0.75 minus λ and set this determinant is equal to 0 and then you solve for the values of λ .

(Refer Slide Time: 19:08)



The image shows a digital whiteboard with handwritten mathematical work. At the top, a 2x2 determinant is set equal to zero: $\begin{vmatrix} 0.75 - \lambda & 0.375 \\ 0.375 & 0.75 - \lambda \end{vmatrix} = 0$. Below this, the equation is simplified to $(0.75 - \lambda)^2 = 0.375^2$. The next step shows the solution for λ : $\lambda = 0.75 \pm 0.375$. Finally, the two eigenvalues are listed in a set notation: $\begin{cases} \lambda_1 = 1.125 \\ \lambda_2 = 0.375 \end{cases}$. At the bottom of the whiteboard, there is a toolbar with various drawing tools and a timer showing 00:00:00.

So, if you do this, you will find that this simply gives an equation of the form 0.75 minus λ square is equal to 0.375 square. Now, if you solve this, the solution is very simple. The λ comes out to be 0.75 plus minus 0.375 whereby you will get λ_1 is equal to 1.125 and λ_2 comes out as 0.375 .

So, these are the 2 Eigen values of the covariance matrix C_x in this particular case and once we have this Eigen values, we have to find out what are the Eigen vectors corresponding to these Eigen values.

(Refer Slide Time: 20:39)

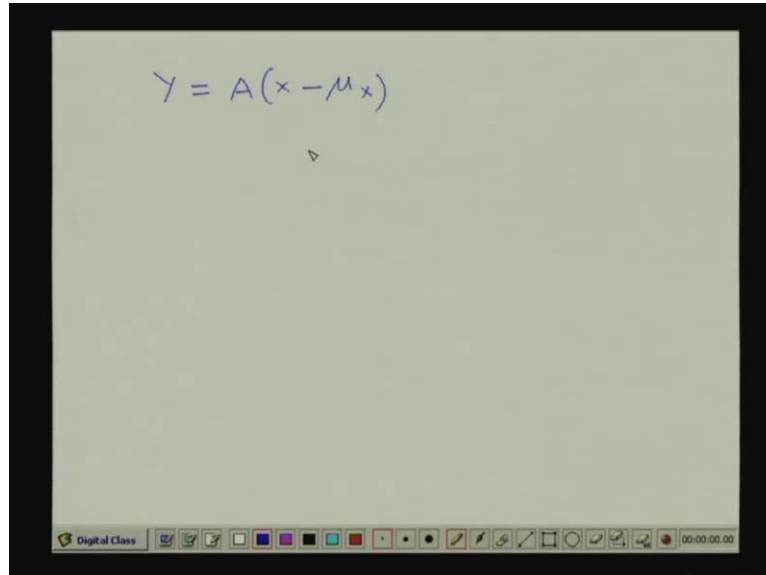
The image shows a digital whiteboard with handwritten mathematical work. At the top, the equation $C_x Z = \lambda Z$ is written. Below it, two eigenvalue equations are shown: $\lambda_1 = 1.125 \Rightarrow e_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $\lambda_2 = 0.375 \Rightarrow e_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$. At the bottom, the transformation matrix A is defined as $A = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. The whiteboard interface includes a toolbar at the bottom with various drawing tools and a timer showing 00:00:00.

And to find out the Eigen vectors, you know that the relation is **for the given matrix** for a given matrix say A or in our particular case, it is C_x , so let us take C_x . So, C_x into say vector Z has to be equal to lambda times Z if Z is the Eigen vector corresponding to the Eigen value lambda and if we solve this, we will find that we get 2 different Eigen vectors corresponding to 2 different Lambda's. So, corresponding to Lambda $_1$ is equal to 1.125, we have the corresponding Eigen vector e_1 which is given as 1 upon root 2 into (1, 1). So, this will be the corresponding Eigen vector.

Similarly, corresponding to the Eigen value lambda $_2$ equal to 0.375, this corresponds to the Eigen vector e_2 which is equal to 1 upon root 2 into 1 minus 1. So, you will find that once we get these Eigen vectors, we can formulate the corresponding transformation matrix. As we said, we will get the transformation matrix A from the Eigen vectors of the covariance matrix C_x but the rows of the transformation matrix A are the Eigen vectors of C_x such that the first row **will correspond to the Eigen vector** will be the Eigen vector corresponding to the maximum Eigen value and the last row will be the Eigen vector corresponding to the minimum Eigen value.

So, in this case, the transformation matrix A will be simply given by 1 upon root 2 to (1, 1, 1, minus 1). Now, what is the implication of this?

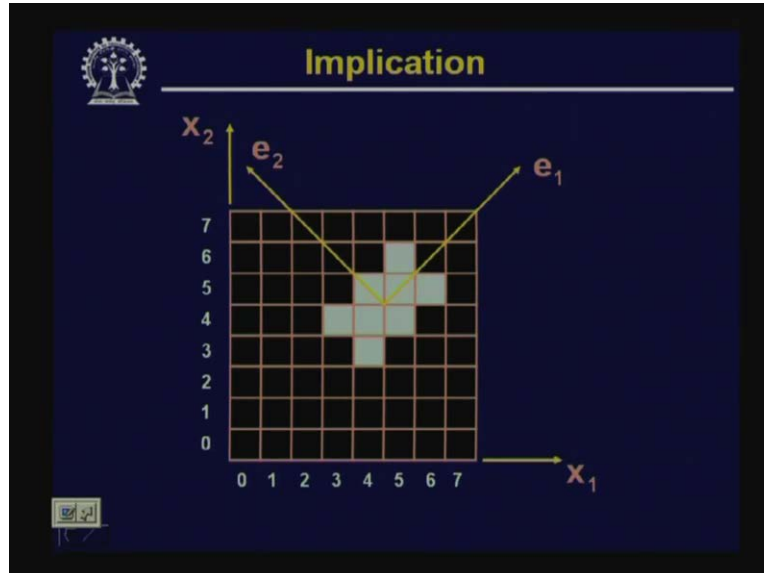
(Refer Slide Time: 23:10)


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

So, you will find that using this particular transformation, transformation matrix; if I apply the K – L transformation, then the transformed output, the transformed vector will by Y equal to A into x minus mu _x.

So, you will find that application of this particular transformation **this particular transformation** amounts to establishing a new coordinate system whose origin is at the centroid of the object pixels. So, this particular transformation K – L transformation, basically establishes a new coordinated system whose origin will be at the center of the object and the axis of this new coordinate system will be parallel to the directions of the Eigen vectors. So, by this what we mean is like this one.

(Refer Slide Time: 24:03)



So, this was our original figure where all the white pixels are the object pixels. Now, by application of this transformation, this $K - L$ transformation with transformation matrix A , we get 2 Eigen vectors. The Eigen vectors are these - e_1 and e_2 . So, you find that this e_1 and e_2 , it forms a new coordinate system and the origin of this coordinate system is located at the center of the object and the axis are parallel to the directions of the vectors e_1 and e_2 and this figure also shows that this is basically a rotation transformation and this rotation transformation aligns the data with Eigen vectors and because of this alignment, different elements of the vector Y , they become uncorrelated.

So, it is only because of this alignment, the data becomes uncorrelated and also because the Eigen values of λ_i appear along the main diagonal of C_y that we have seen earlier, this λ_i basically tells the variance of the component Y_i along the Eigen vector e_i and later on we will see the application of this kind of transformation to align the objects along the Eigen vectors and this is very very important for object recognition purpose.

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 $K - L$ transformation basically aligns the data along the Eigen 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 state of vectors y from another state of vectors x using the transformation matrix A where A was derived using the Eigen vectors of the covariance matrix of x that is C_x .

(Refer Slide Time: 26:40)

$$Y = A(x - \mu_x)$$
$$A^{-1} = A^T$$
$$X = A^T Y + \mu_x$$

$A_k \rightarrow k$ no. of eigen vectors.

$A_k \Rightarrow k \times n$

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

So, our K – L transformation expression was Y equal to A into x minus μ_x . Now, here you find that because this matrix A, the rows of this matrix A are the Eigen vectors 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 is 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 it. 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 μ_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 this matrix A has been formed by using all the Eigen vectors 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 Eigen vectors of the covariance matrix C_x . Rather, I will consider say k number of Eigen vectors and using that k number of Eigen vectors, I will make a transformation matrix say A_k .

So, this A_k is formed using k number of Eigen vectors **k number of Eigen vectors** of matrix C_x . I am not considering all the Eigen vectors of the matrix C_x and obviously because I am taking k number of Eigen vectors, I will take those Eigen vectors corresponding to k-largest Eigen values. 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 μ_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 Eigen vectors; 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: 31:06)

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

$A_k \Rightarrow k \times n$
 $y = k$
 $A_k^T \Rightarrow n \times k$

$$\hat{x} \rightarrow n$$

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

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

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 μ_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 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 will 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_{ms} is given by sum of λ_j where j varies from 1 to n minus sum of λ_i where i varies from 1 to k which is nothing but sum of λ_j where j varies from k plus 1 to n.

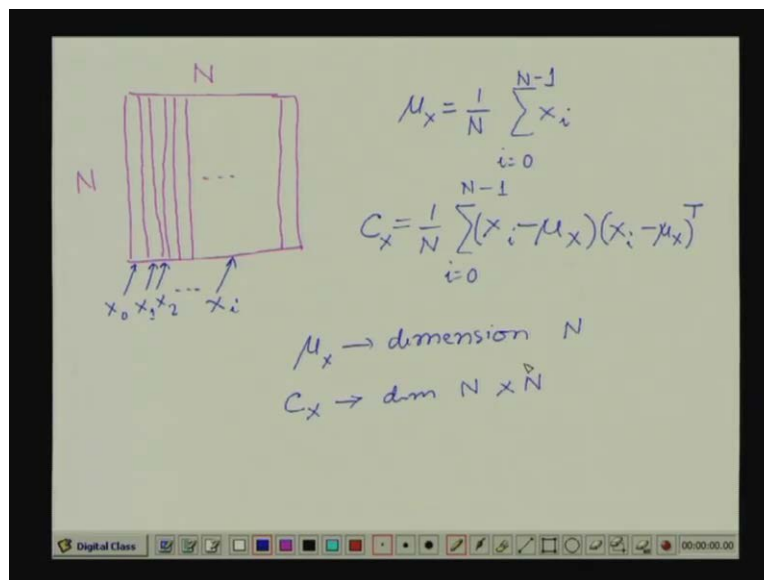
So, you will find that this mean square error, this term that we have got; you remember that while forming our transformation matrix A_k , we have considered k number of Eigen vectors of matrix C_x and these k number of Eigen vectors **corresponding to** corresponds to largest Eigen

values of matrix C_x and in this particular expression, the mean square error is given by the sum of those Eigen values whose corresponding Eigen vectors was not considered for formation of our transformation matrix A and because the corresponding Eigen values are the smallest Eigen values, so this particular transformation and the corresponding inverse transformation ensures that the mean square error of the reconstructed signal or the mean square error between x and \hat{x} will be minimum.

That is because of this summation consists of summation of only those Eigen values which are having the minimum value. So, that is why this $K - L$ transform is often called an optimum transform because it minimizes the error of reconstruction between x and \hat{x} . Now, these 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 that 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 $K - L$ transform over an image?

(Refer Slide Time: 35:50)



So, we have a digital image is a 2 dimensional array of quantized intensity values. So, a digital image as it is represented by a 2 dimensional array of quantized 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 1 ways.

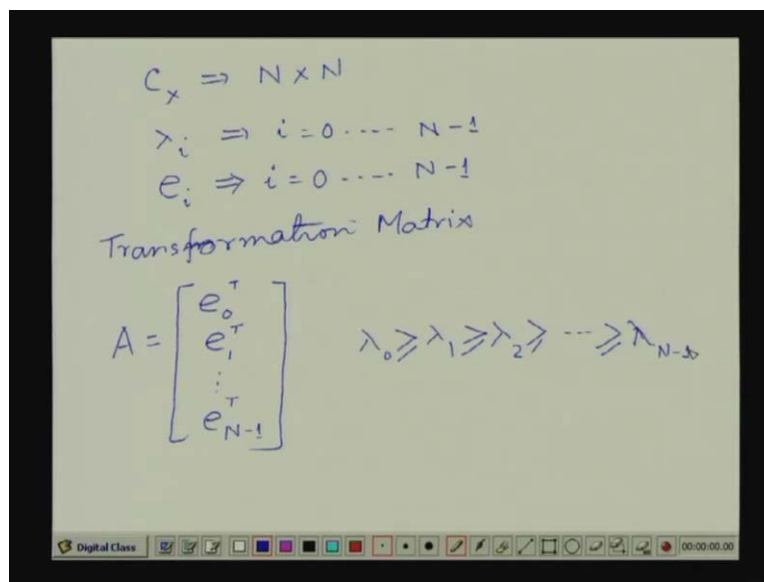
So, let us assume in this particular case that we represent every column of this 2 dimensional array as a vector. So, if we do that then every column of this; so this will be represented by a

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 this n number of vectors; for this n number of vectors, we can find out the mean vector which is μ_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 had already seen that this is 1 upon capital N summation x_i minus μ_x into x_i minus μ_x transpose where this i will vary from 0 to capital N minus 1 .

And, here you will find that our mean vector μ_x , this is of dimension capital N whereas the covariance matrix C_x , this is of dimension capital N by capital N . So, once we have obtained the mean vector μ_x and the covariance matrix C_x , we can find out the Eigen vectors and Eigen values of this covariance matrix C_x and as we have already seen

(Refer Slide Time: 39:33)



that because this particular covariance matrix C_x is of dimension capital N by capital N , there will be N number of Eigen values λ_i where this i varies from 0 to capital N minus 1 and corresponding to every Eigen value λ_i , there will be an Eigen vector e_i . So, this e_i , Eigen vector e_{ii} here again will be vary from 0 to capital N minus 1 .

Now, given this n number of Eigen vectors, for n number of Eigen values; 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 Eigen vector 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 Eigen vectors 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 Eigen vectors, so I will have e_N minus 1 transpose where this e_0 corresponds to the Eigen value λ_0 and obviously in this case, as we have already said that our assumption is λ_0 is greater than or equal to λ_1 which is greater than or equal to λ_2 and continued like this, it is greater than or equal to λ_{N-1} . So, this is how we form the transformation matrix A .

Now, from this transformation matrix, we can make a truncated transformation matrix where instead of using all the Eigen vectors of the covariance matrix C_x , we consider only the first k number of Eigen vectors which corresponds to k number of Eigen values, k number of the largest Eigen values.

(Refer Slide Time: 42:17)

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

$$Y_i = A_k (x_i - \mu_x); \quad i=0, 1, \dots, N-1$$

$\begin{matrix} \swarrow & \searrow \\ k \times N & N \times 1 \end{matrix}$

$Y_i \rightarrow k \times 1$

$k \times N$

So, we form the transformation matrix, the modified transformation matrix A_k using the first k number of Eigen vectors. So, in our case, A_k will be e_0 transpose, e_1 transpose and likewise it will go upto e_{k-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 μ_x where this i varies from 0 to capital N minus 1. So, here you find that because **A_k is of dimension** the dimension of A_k is k by N and dimension of X_i and μ_x both of them are of dimension N by 1; so X_i minus μ_x , this is a vector of dimension capital N by 1. So, this particular vector, this is of dimension capital N by 1.

So, you will find that when I multiply, when I perform this transformation - A_k into X_i minus μ_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 transformed 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 consist 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 of 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 perfectly reconstructed image, rather what we will get is an approximate image.

(Refer Slide Time: 45:46)

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

$A_k \rightarrow$ needs to be saved.

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

So, this approximate image will be given by \hat{x}_i that is equal to A_k transpose Y_i plus μ_x where this \hat{x}_i here you find that it will be of dimension capital N. So, collection of all these \hat{x}_i 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 Eigen values which are left out; corresponding to which the Eigen vectors were not considered for formation of the transformation matrix.

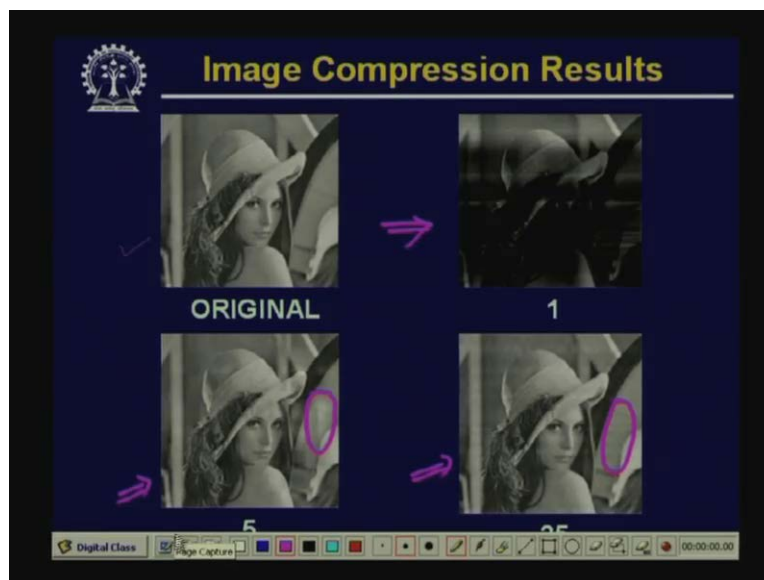
So here, you find that because now in this case for getting the reconstructed image, what are the quantities that we have to save? Obviously, the first quantity that we have to save, the first information that we have to save is the transformation matrix A_k . So, this A_k needs to be saved and the other information that you 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 2 quantities A_k and the set of transformed vectors Y_i , then from these 2 quantities we can reconstruct an approximate original image given by the vectors \hat{x}_i . So, you will 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 Eigen vectors 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 considered only 1 Eigen vector to form our transformation matrix A . It can be 2 where we consider only 2 Eigen vectors to form the transformation matrix A and depending upon the number of the Eigen vectors, the amount of compression 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: 49:00)

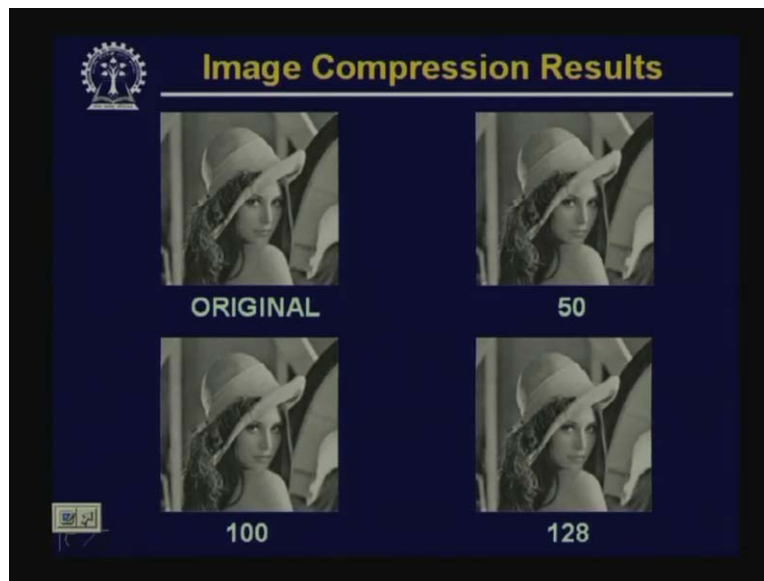


So here, you find that we have shown some of the images. Here, the top image, this is the original image. Now, when this original image is actually transformed and reconstructed using the transformation matrix with only 1 Eigen vector; so this the Eigen vector which **corresponds to the** corresponds to the largest Eigen value of the covariance matrix. Then the reconstructed image that we get is given by this result. So, here we 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 Eigen vectors in the transformation matrix; when I use 5 Eigen vectors as the transformation matrix, then the reconstructed image is given by this one. You will 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 Eigen vectors further, that is we use 25 Eigen vectors to form the transformation matrix; then this is the reconstructed imaged that we get.

Now, if you closely observe between these 2 images, compare these 2 images; you will find that there are some artifacts. See for example, in this particular region, there is an artifact, 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 Eigen vectors from 5 to 35.

(Refer Slide Time: 51:41)



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

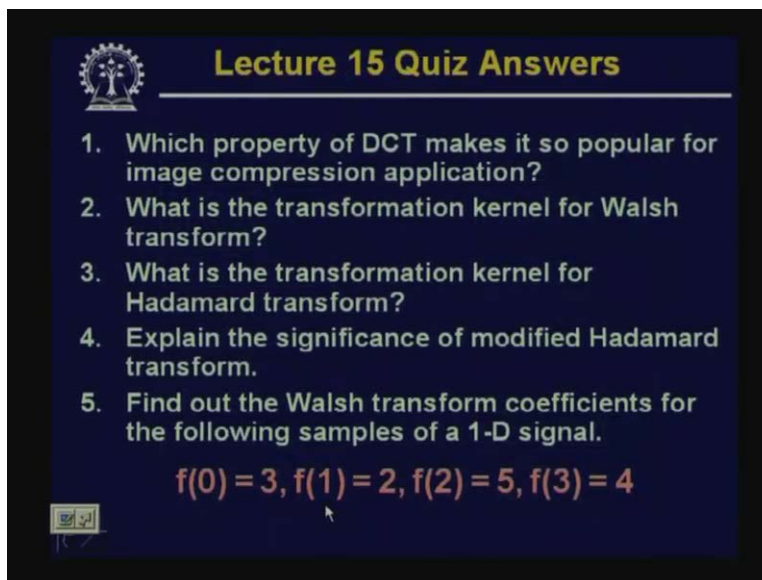
So here, we have discussed about the $K - L$ transformation where we have said the $K - L$ transformation is fundamentally defined from the other transformations that we have discussed earlier that is the discrete Fourier transformation, discrete cosine transformation and so on and there we have said that in those case of transformations, the transformation matrix or the transformation kernel is fixed whereas in case of $K - L$ transformation, you will 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 dependent upon the data, it is not fixed. So, that is the fundamental difference between the other transformations with the $K - L$ transformation. But the advantage of the $K - L$ transformation that is quite obvious from the reconstructed images is 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 in the earlier result that we have shown where using only 1 Eigen vector as the transformation matrix, this particular result, here, using only 1 Eigen vector 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, it shows that the energy compaction in a Eigen vector, in the number of components is much higher in case of K – L transform than in case of other transformation.

But 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 its strong property of energy compaction, K – L transformation has not been much popular for data compression operations. With this, we come to the end of our discussion on transformations. Now, let us see the answers to our previous days lecture.

(Refer Slide Time: 54:16)



The image shows a slide titled "Lecture 15 Quiz Answers" with a list of five questions and a sample signal. The slide has a dark blue background with yellow and white text. A small logo is in the top left corner. The questions are:

1. Which property of DCT makes it so popular for image compression application?
2. What is the transformation kernel for Walsh transform?
3. What is the transformation kernel for Hadamard transform?
4. Explain the significance of modified Hadamard transform.
5. Find out the Walsh transform coefficients for the following samples of a 1-D signal.

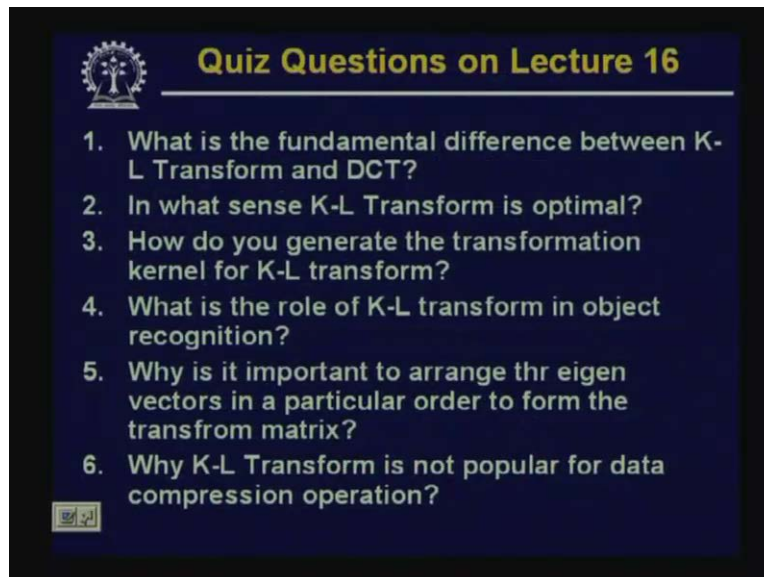
The sample signal is given as $f(0) = 3, f(1) = 2, f(2) = 5, f(3) = 4$.

So here, it is quite obvious that property of the DCT which makes it popular for image compression applications is the energy compaction operation, energy compaction property. The transformation kernels for Walsh transformation, Hadamard transformation has already been discussed during our lecture.

Significance of modified Hadamard transform: so here, we have said that by modified Hadamard transform I mean the ordered Hadamard transform. The significance of ordered Hadamard transform is that here we can correlate the sequence of the signal with the variable u which is done in case of DFT or in case of DCT where the increasing value of u means increasing value of frequency components and here equivalent to the frequency component in case of Hadamard transform, we have defined what is called sequence and the increasing value of u should indicate increasing value of sequence and that is basically the significance of this Hadamard transform and we have seen during our last lecture that because of this, here also we can get some energy compaction property that is most of the energy is confined within few Hadamard coefficients.

The fifth problem, it was a problem where you have to find out the Walsh transformation coefficients of the following samples. This also quite simple from the discussion that we had; if i simply replace these values in the Walsh transform expressions that we discussed, then we will get the Walsh transform coefficients.

(Refer Slide Time: 55:56)



Now, coming to today's lecture, questions on today's lecture: the first question is what is the fundamental difference between K – L transform and discrete cosine transform. The second question, in what sense K – L transform is optimal?

Third question is how do you generate the transformation kernel for K – L transform? Fourth question, what is the role of K – L transform in object recognition? Fifth one, why is it important to arrange the Eigen vectors in a particular order to form the transform matrix and the last question why K – L transform is not popular for data compression operation?

Thank you.