Computer Vision and Image Processing - Fundamentals and Applications

Professor Dr. M.K. Bhuyan
Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati, India

Lecture 11

Image Transforms

Welcome to the NPTEL MOOCs course on Computer Vision and Image Processing – Fundamentals and Application. Already I have defined mathematically the image, the image is represented by f (x, y), f(x, y) means intensity at a particular point the point is x ,y that is called the spatial domain representation of an image. In an image, we have frequency information. Suppose, if I consider edges or a boundary that is nothing but the high-frequency information.

In edges in the boundary, there is an abrupt change of grayscale intensity value, that is why the high-frequency information is present in the edges in the boundary. If I consider the constant intensity region, the homogeneous region that corresponds to the low-frequency information.

So, I can convert the spatial domain information into the frequency domain information for better analysis of an image, I can do some transformation, I can apply the DFT the discrete Fourier transform, I can apply the DCT the discrete cosine transform like this so that the spatial domain information can be converted into a frequency domain information.

In the case of the frequency, what is the definition of the frequency? Frequency means, spatial rate of change of grayscale value that is the definition of frequency. Now, in this transformation, that signal is represented as a vector and the transformation changes the basis of the signal space, that is the definition of the image transformation.

And the transformation is quite useful for compact representation of data. So, that means, the image can be represented by using few transformation coefficients. So, I can apply DFT, I can apply Discrete Cosine transform and the image can be represented by using transform coefficients that means, the compact representation of an image that is the image transformation.

And because of this transformation, it is easy to calculate convolution or maybe the correlation I can compute because convolution means in the frequency domain it is nothing but the multiplication, in spatial domain it is the convolution. So, because of this transformation, I can easily do convolution and let us see what is the meaning of the image transformation. So, in my next slide, I can explain the concept of image transformation.

(Refer Time Slide: 03:01)

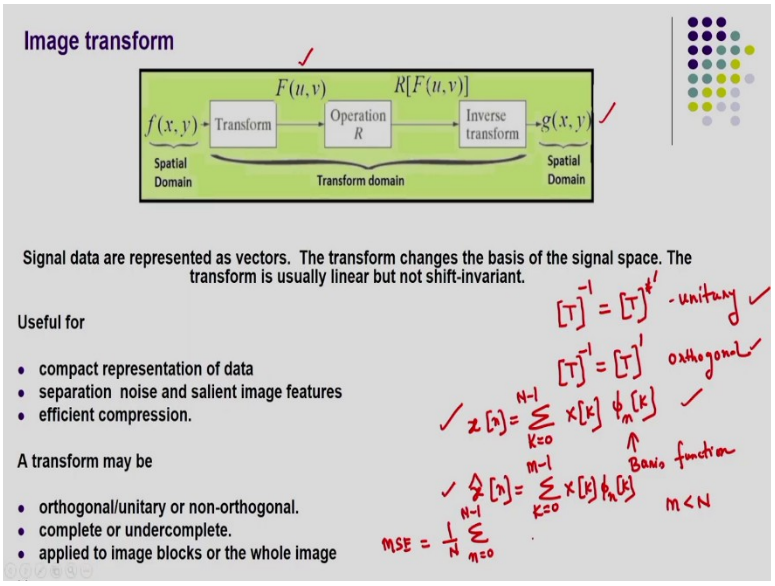


Image transform

Signal data are represented as vectors. The transform changes the basis of the signal space. The transform is usually linear but not shift-invariant.

Useful for

- compact representation of data
- separation noise and salient image features
- efficient compression.

A transform may be

- orthogonal/unitary or non-orthogonal.
- complete or undercomplete.
- applied to image blocks or the whole image

$[T]^{-1} = [T]^{*'}$ - unitary ✓

$[T]^{-1} = [T]'$ orthogonal ✓

✓ $x[n] = \sum_{k=0}^{N-1} x[k]\, \phi_n[k]$ ✓

$\uparrow$ Basis function

✓ $\hat{x}[n] = \sum_{k=0}^{m-1} x[k]\, \phi_n[k]$ $m < N$

$MSE = \frac{1}{N} \sum_{n=0}^{N-1}$

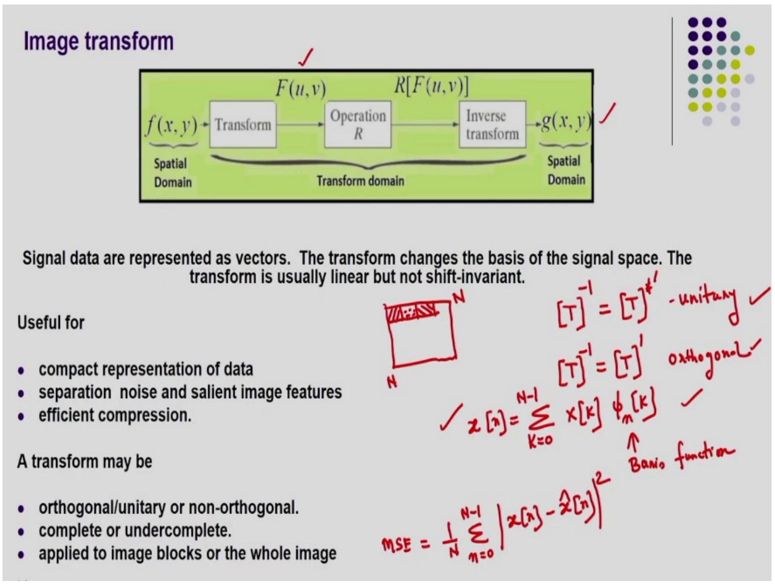


Image transform

Signal data are represented as vectors. The transform changes the basis of the signal space. The transform is usually linear but not shift-invariant.

Useful for

- compact representation of data
- separation noise and salient image features
- efficient compression.

A transform may be

- orthogonal/unitary or non-orthogonal.
- complete or undercomplete.
- applied to image blocks or the whole image

$[T]^{-1} = [T]^{*'}$ - unitary ✓

$[T]^{-1} = [T]'$ orthogonal ✓

✓ $x[n] = \sum_{k=0}^{N-1} x[k]\, \phi_n[k]$ ✓

$\uparrow$ Basis function

$N \times N$

$MSE = \frac{1}{N} \sum_{n=0}^{N-1} \left| x[n] - \hat{x}[n] \right|^2$

So, in this block diagram you have seen the input is f (x, y) that is the image, after this, we are doing some transformation. Operator transformation I am getting F (u, v), so F (u, v) is nothing but that is the transformed image, u is the spatial frequency along the x-direction and v is the spatial frequency along the y-direction.

After this, we can do some operations in the frequency domain that is the operation R I am doing in the frequency domain and after this, I am doing inverse transformation. So, that we will get the spatial domain information after processing. So, I am getting g (x, y). So, in the case of the inverse transformation signal data that is represented as vectors and the transformation changes the basis of the signal space.

And one important point is the transformation is usually linear but not shift-invariant. And already I had explained that it is useful for a compact representation of data and because of these transformations, I can separate noise and the salient image features because the image is represented by few transform coefficients.

So, it is easy to separate noise and the salient image features, and also it is useful for image compression because the image I am considering by only considering few coefficients that is which are more important than we are considering and neglecting the remaining coefficients, so like this, we can do the image compression. And the transform may be orthogonal or maybe the non-orthogonal.

So, suppose if I consider one transformation matrix, the transformation matrix is T that is a complex matrix and if I take that T inverse that is equal to T complex conjugate transpose then this matrix T is called the unitary matrix. It is called the unitary matrix and in this case, the transformation is called the unitary transformation. And suppose that T is real, the transformation matrix is real then T inverse is equal to T transpose then in this case and this matrix is called the orthogonal matrix.

So, I have defined unitary matrix and the orthogonal matrix. Based on this condition my transformation maybe unitary transformation or maybe the orthogonal transformation. So, transformation may be orthogonal if this condition is satisfied or maybe the unitary and if this condition is not satisfied then in this case it will be non-unitary or maybe the non-orthogonal transformation.

One definition is the transformation may be complete or under complete. What is the meaning of this? Suppose, let us consider x n, I am representing the input data x n is represented like this, N minus 1 K is equal to 0 to N minus 1, X K and phi n K and this is the basis function. So, my input data that is input vector is represented like this x n is equal to X K that means, I am doing some transformation X K and the basis function is this.

So, in this case, K is equal to 0 to N minus 1. This x n can be approximately represented like this, this is the approximate representation of x n. So, K is equal to 0 to M minus 1, now it is M minus 1, X K phi n K. So, in the second representation that is the approximate representation I am only considering K is equal to 0 to M minus 1. In the first case I am considering K is equal to 0 to N minus 1 that means, I am considering the n number of coefficients that means, x0, x1 x2 like this I am considering.

But in the second case I am considering only M number of coefficients then in this case that your M is less than N. Now if I consider the first one, that my transformation will be the complete transformation, and if I considered the second case that is the under complete case and in both the cases if I see, I can determine the mean square error is nothing but 1 by N summation n is equal to 0 to N minus 1.

And in this case the mean square error also I can determine, so a transformation may be complete or under complete, so you can understand this. And another one is the transformation can be applied for the whole image that means, if I consider a whole image, this is the whole image, this N-by-N image.

So, for the entire image, I can apply to the transformation, or otherwise, I can apply the block by block. So, suppose if I consider is the block, one block I am applying the transformation, next block I am applying the transformation like this I can apply the transformation block by block. So, that the information may be applied to image blocks or maybe to the whole image.

So, this is the definition of the image transformation and the transformation may be orthogonal or the non-orthogonal, complete or under complete, applied to the image blocks or the whole image. Now, based on this orthogonality concept, I can show you a different type of transformation. So, what are the different types of the transformation?

(Refer Time Slide: 09:44)

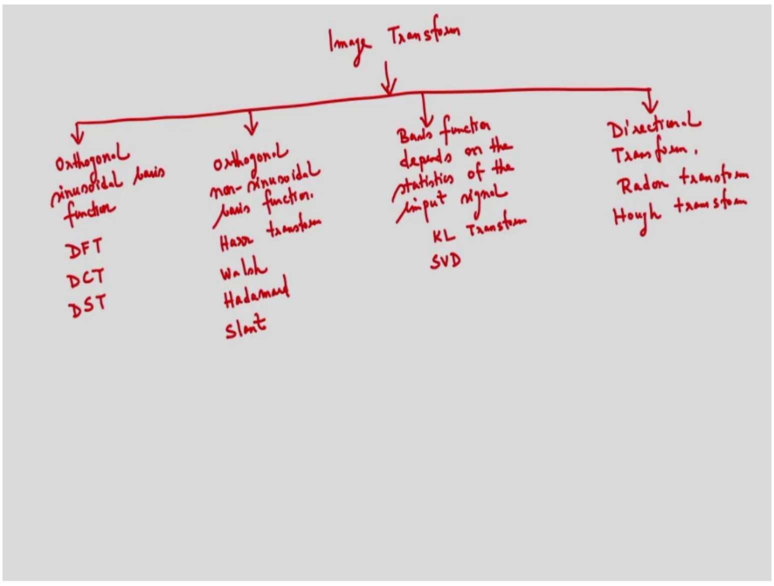


Image Transform
Orthogonal sinusoidal basis function
DFT
DCT
DST
Orthogonal non-sinusoidal basis function.
Haar transform
Walsh
Hadamard
Slant
Basis function depends on the statistics of the input signal
KL Transform
SVD
Directional Transform.
Radon transform
Hough transform

So, images permission, so first one is I am considering orthogonal sinusoidal basis function. The first case is orthogonal sinusoidal basis function, the second case is orthogonal non-sinusoidal basis function, or I can consider the basis function depends on the statistics of the input signal and another one is the directional transformation. So, I can classify image transformation like this, the first one is orthogonal sinusoidal basis function. My transformation function is orthogonal and also, I am considering the sinusoidal basis function.

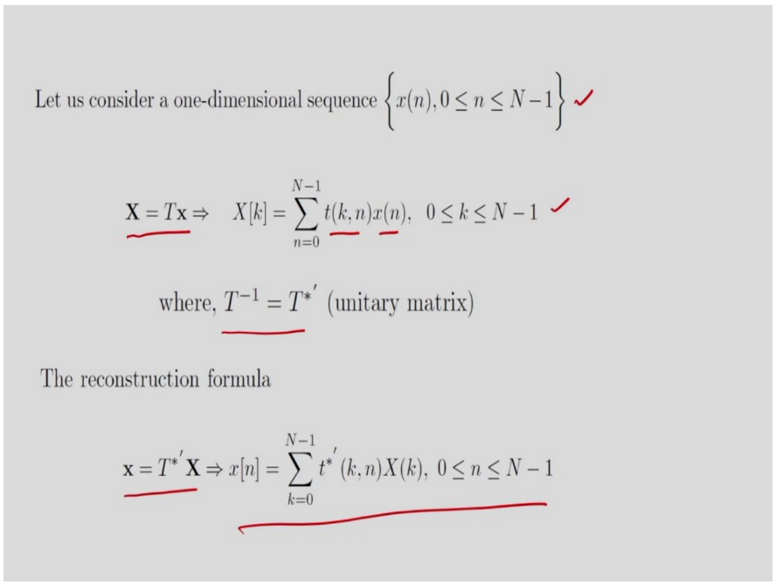
So, examples like the DFT the Discrete Fourier Transform, or the DCT the Discrete Cosine Transform, or maybe the Discrete Sine Transform DST these are the examples. So, basis function is orthogonal and also the sinusoidal function. Another one is orthogonal non-sinusoidal basis function. So, I can give some examples like the Harr transformation, Harr transform that is we will discuss in the wavelet transform.

In the wavelet transform you can understand what is Harr transformation, another transform like Walsh transform, Hadamard transform, or maybe slant transform. In this case, basis function is orthogonal but, in this case, it is non-sinusoidal basis function. Another case, the next one is the basis function depends on the statistics of the input data. Then in this case, I can give you one example one is the KL transformation, I will discuss about the KL transform and another one is singular value decomposition SVD, so I can give these examples.

So, basis function is not fixed, but basis function depends on the statistics of the input data, the input signal that is the examples like the KL transform and singular value decomposition. And regarding the directional transformation, so already I have discussed about the Radon transform that is one example is the Radon transform and another example I can give I will discuss later on that is the Hough transform.

So, I have these types of transformations one is the DFT, DCT, and DST that is the orthogonal sinusoidal basis function and another one is the Harr transformation, Walsh transformation, Hadamard transformation, Slant transformation orthogonal non-sinusoidal basis function and another one is the basis function depends on the statistics of the input signal that is the KL transform and the SVD and directional transformation like Radon transform and the Hough transform.

(Refer Time Slide: 14:21)



Let us consider a one-dimensional sequence $\left\{x(n), 0 \le n \le N-1\right\}$ ✓

$$\mathbf{X} = T\mathbf{x} \Rightarrow \quad X[k] = \sum_{n=0}^{N-1} t(k,n)x(n), \quad 0 \le k \le N-1$$ ✓

where, $T^{-1} = T^{*'}$ (unitary matrix)

The reconstruction formula

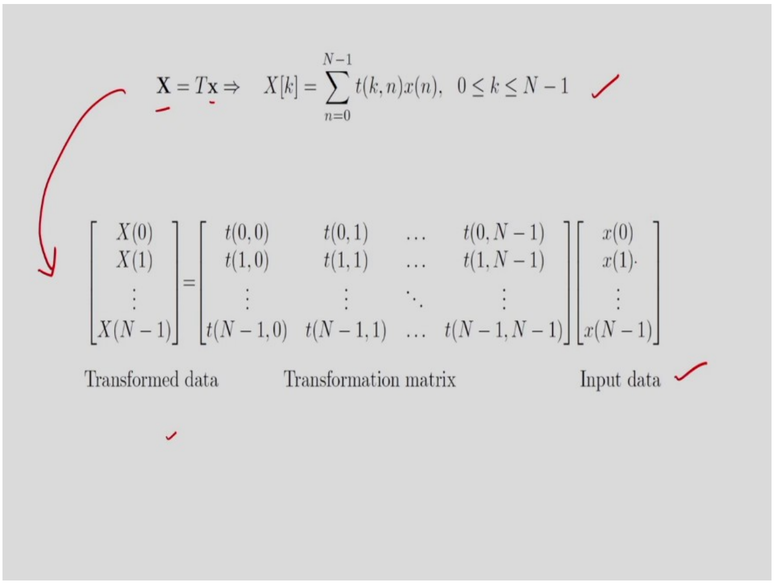
$$\mathbf{x} = T^{*'}\mathbf{X} \Rightarrow x[n] = \sum_{k=0}^{N-1} t^{*'}(k,n)X(k), \; 0 \le n \le N-1$$

Now, let us consider one-dimensional sequence, the one-dimensional sequence is x n. So, it is n is from 0 to N minus 1. Now, I am doing some transformation, the transformation is x is equal to T x, so this transformation I am considering. This x is the, small x is the input data and in this case, the X capital is equal to t x, t is the transformation matrix. So, I can write like this X k is equal to t k n, t k n is this, this is the transformation kernel x n is the input data, the input sequence.

And in this case, if I consider a transformation is something like this the T inverse equal to T complex conjugate transpose. So, already I have defined this is called the unitary matrix and the transformation will be the unitary transformation. And for the reconstruction I can use this formula that is reconstruction of the original sequence small x is equal to T complex conjugate transpose X because the T complex conjugate transpose is nothing but T inverse.

So, in this case, the reconstruction formula I can get this, this is the reconstruction formula. So, x n is equal to k is equal to 0 to N minus 1 t complex conjugate transpose k comma n X k and the N is from 0 to N minus 1.

(Refer Time Slide: 15:44)



$$\mathbf{X} = T\mathbf{x} \Rightarrow \quad X[k] = \sum_{n=0}^{N-1} t(k,n)x(n), \quad 0 \le k \le N-1$$

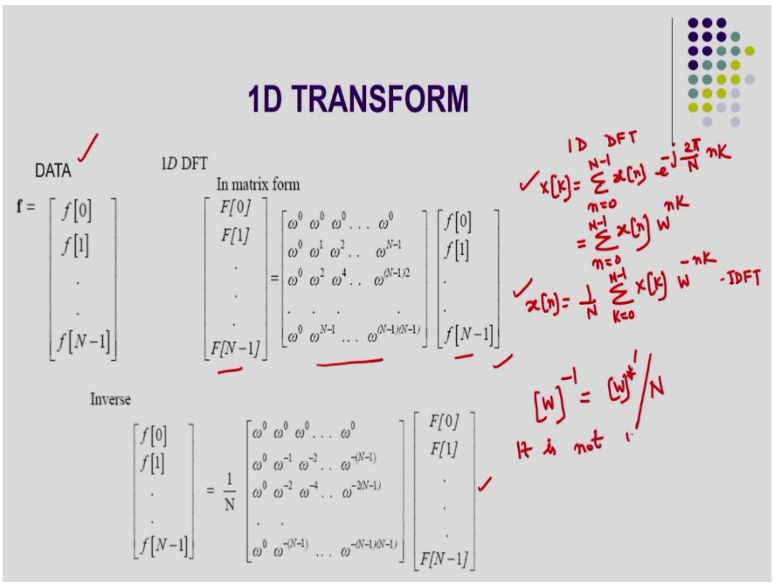
$$\begin{bmatrix} X(0) \\ X(1) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} t(0,0) & t(0,1) & \dots & t(0,N-1) \\ t(1,0) & t(1,1) & \dots & t(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ t(N-1,0) & t(N-1,1) & \dots & t(N-1,N-1) \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

Transformed data Transformation matrix Input data

Now, let us consider this case, the X is equal to T x, so X capital X and that is the transform data, T is the transformation and x is the input sequence, so already I have defined this one. So, in the matrix from I can represent in this form, the first one is the transform data. So, this equation I am writing in the matrix from.

So, the first one is the transform data and after this I am considering the transformation matrix and after this, I am considering the input data. In this case that x naught the small x naught, x1, xn minus 1 that is the input data and x naught, x1, xn minus 1 that is the transform data there is a capital and I am considering that this permission matrix.

(Refer Time Slide: 16:34)



1D TRANSFORM

DATA

$$\mathbf{f} = \begin{bmatrix} f[0] \\ f[1] \\ \cdot \\ \cdot \\ f[N-1] \end{bmatrix}$$

1D DFT

In matrix form

$$\begin{bmatrix} F[0] \\ F[1] \\ \cdot \\ \cdot \\ F[N-1] \end{bmatrix} = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 \dots & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 \dots & \omega^{N-1} \\ \omega^0 & \omega^2 & \omega^4 \dots & \omega^{(N-1)2} \\ \cdot & \cdot & \cdot & \cdot \\ \omega^0 & \omega^{N-1} & \dots & \omega^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} f[0] \\ f[1] \\ \cdot \\ \cdot \\ f[N-1] \end{bmatrix}$$

Inverse

$$\begin{bmatrix} f[0] \\ f[1] \\ \cdot \\ \cdot \\ f[N-1] \end{bmatrix} = \frac{1}{N} \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 \dots & \omega^0 \\ \omega^0 & \omega^{-1} & \omega^{-2} \dots & \omega^{-(N-1)} \\ \omega^0 & \omega^{-2} & \omega^{-4} \dots & \omega^{-2(N-1)} \\ \cdot & \cdot & & \\ \omega^0 & \omega^{-(N-1)} & \dots & \omega^{-(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} F[0] \\ F[1] \\ \cdot \\ \cdot \\ F[N-1] \end{bmatrix}$$

1D DFT

$$X[K] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}nK}$$

$$= \sum_{n=0}^{N-1} x[n]\, W^{nK}$$

$$x[n] = \frac{1}{N}\sum_{K=0}^{N-1} X[K]\, W^{-nK} \quad \text{- IDFT}$$

$$[W]^{-1} = [W]^{*'}/N$$

It is not

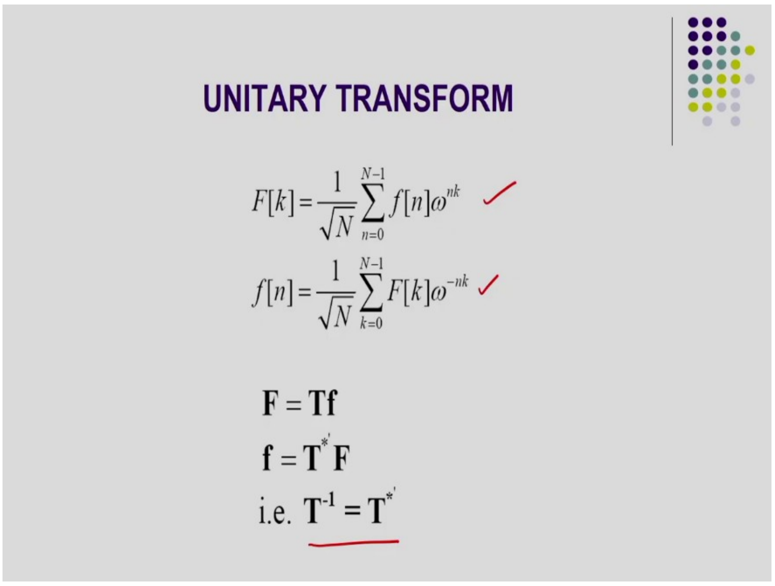
Now, let us consider 1D transformation. So, I think already you know the DFT, so how to write a DFT. So, 1D DFT I am considering first, 1D DFT. So, one-directional DFT I can write like this X K is equal to n is equal to 0 to N minus 1 x n e to the power minus j twice pi divided by N n k. That is the I can write like this n is equal to 0 to N minus 1 x n W n K.

So, in this case, W is the twiddle factor, you know W is the twiddle factor DFT and x n I can write like this, x n also I can write x n is nothing but 1 by N, K is equal to 0 to N minus 1, K is equal to 0 to N minus 1 and it is X K W minus n k, so I can write like this, this is a 1D DFT. So, in this case, I am considering the data vector, these are the data vector and I am considering the 1D DFT in the matrix form, so 1D DFT into matrix form I can write like this.

So, this is the DFT already I have defined the DFT, 1d DFT I have defined and in the matrix from I can represent like this. So, this is my transform data and this is the transformation matrix and this is my input data. And regarding the inverse one, if I consider the inverse DFT, so the inverse DFT is nothing but this, this is the inverse DFT, this is the IDFT, so inverse DFT I can represent like this.

And in this case, if I see this transformation, the transformation matrix is suppose W, W inverse is equal to W complex conjugate transpose and divided by n that means, as part of definition of the unitary matrix the DFT matrix is not unitary, so it is not unitary the DFT matrix is not unitary. So, I can make the DFT matrix unitary, if you see my second slide you can understand this one.

(Refer Time Slide: 19:14)



UNITARY TRANSFORM

$$F[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f[n] \omega^{nk} \checkmark$$

$$f[n] = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} F[k] \omega^{-nk} \checkmark$$

$$\mathbf{F} = \mathbf{T}\mathbf{f}$$

$$\mathbf{f} = \mathbf{T}^{*'}\mathbf{F}$$

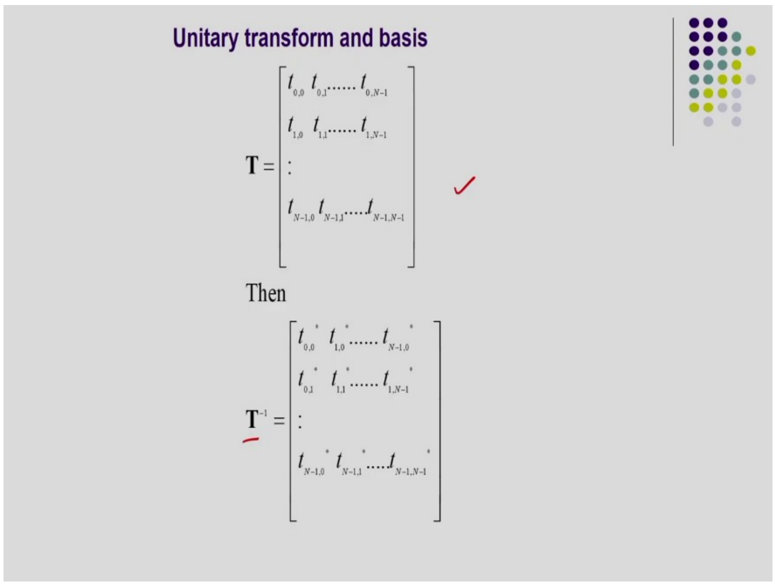
i.e. $\underline{\mathbf{T}^{-1} = \mathbf{T}^{*'}}$

So, I am defining the DFT like this now. So, DFT is F k 1 by root N, f n w nk and this is my inverse DFT. So, if I consider this then in this case my transformation matrix will be like this. So, T inverse is equal to T complex conjugate transpose that means, in this case I am getting the unitary DFT. So, this discussion up to this discussion you have understood that the meaning of the unitary transformation and the orthogonal transformation.

In the case of the unitary transformation, if I consider a unitary matrix that means, the T inverse is equal to T complex conjugate transpose that is the definition of the unitary matrix. And suppose the matrix T is the real matrix then the T inverse is equal to T transpose that is the definition of the orthogonal matrix and based on this I may have unitary transformation or orthogonal transformation.

And in this example, I have shown the DFT transformation, the DFT transformation is not unitary but I can make it unitary. I have defined the DFT like this DFT is 1 by root N like this I have defined so that the DFT transformation matrix that will be unitary. So, that is the definition of the unitary transformation and the orthogonal transformation.

(Refer Time Slide: 20:36)



Unitary transform and basis

$$\mathbf{T} = \begin{bmatrix} t_{0,0} & t_{0,1} \ldots\ldots & t_{0,N-1} \\ t_{1,0} & t_{1,1} \ldots\ldots & t_{1,N-1} \\ \vdots & & \\ t_{N-1,0} & t_{N-1,1} \ldots\ldots & t_{N-1,N-1} \end{bmatrix} \quad ✓$$

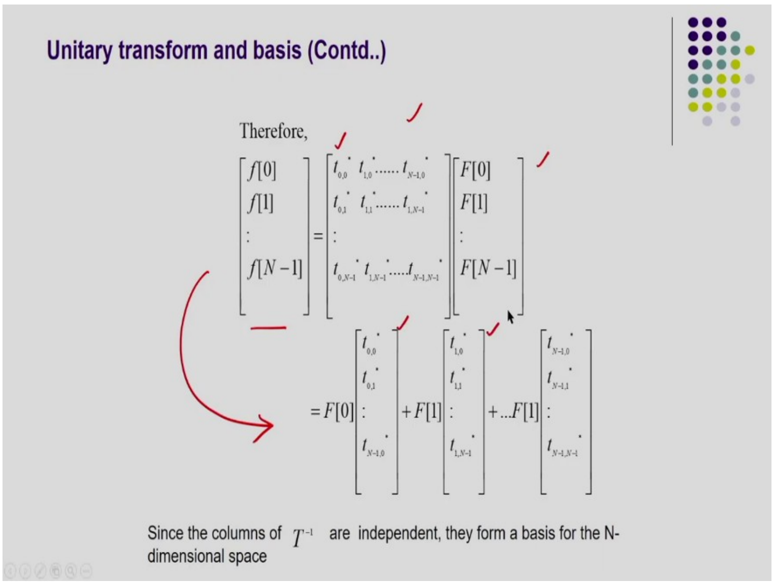
Then

$$\mathbf{T}^{-1} = \begin{bmatrix} t_{0,0}^{*} & t_{1,0}^{*} \ldots\ldots & t_{N-1,0}^{*} \\ t_{0,1}^{*} & t_{1,1}^{*} \ldots\ldots & t_{1,N-1}^{*} \\ \vdots & & \\ t_{N-1,0}^{*} & t_{N-1,1}^{*} \ldots\ldots & t_{N-1,N-1}^{*} \end{bmatrix}$$

Now, let us consider the unitary transformation and basis. So, I am considering one transformation matrix that transformation matrix is T and I am considering the unitary transformation. So, transformation matrix is the rules are like this t00, t01, t0 n minus 1, 310 like this I have the transformation matrix. Then what will be my T inverse?

So, I am calculating the T inverse, T inverse is very easy to calculate because it is the unitary matrix. So, in this case, I have to do complex conjugate and after this, I am doing the transpose, the transpose of the matrix. So, T inverse is equal to T complex conjugate and transpose. So, I am getting T inverse.

(Refer Time Slide: 21:18)



Unitary transform and basis (Contd..)

Therefore,

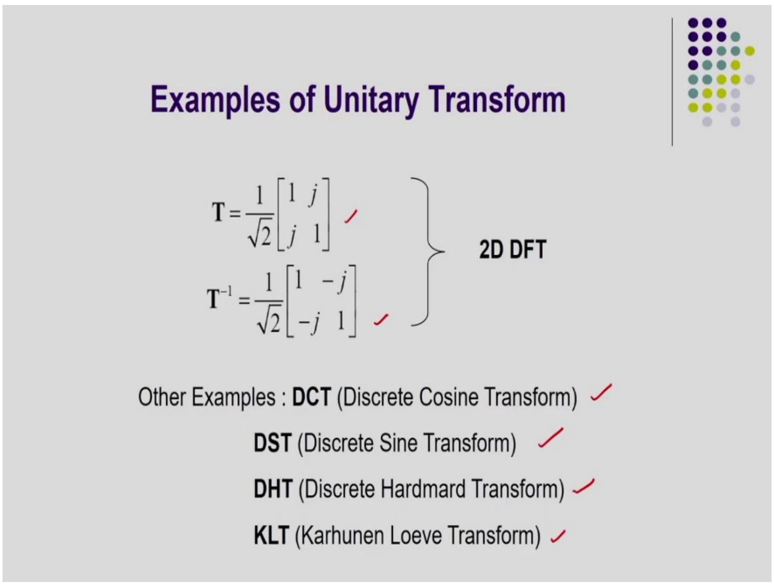
$$\begin{bmatrix} f[0] \\ f[1] \\ \vdots \\ f[N-1] \end{bmatrix} = \begin{bmatrix} t_{0,0}^{*} & t_{1,0}^{*} \cdots\cdots & t_{N-1,0}^{*} \\ t_{0,1}^{*} & t_{1,1}^{*} \cdots\cdots & t_{1,N-1}^{*} \\ \vdots & & \\ t_{0,N-1}^{*} & t_{1,N-1}^{*} \cdots\cdots & t_{N-1,N-1}^{*} \end{bmatrix} \begin{bmatrix} F[0] \\ F[1] \\ \vdots \\ F[N-1] \end{bmatrix}$$

$$= F[0] \begin{bmatrix} t_{0,0}^{*} \\ t_{0,1}^{*} \\ \vdots \\ t_{N-1,0}^{*} \end{bmatrix} + F[1] \begin{bmatrix} t_{1,0}^{*} \\ t_{1,1}^{*} \\ \vdots \\ t_{1,N-1}^{*} \end{bmatrix} + \ldots F[1] \begin{bmatrix} t_{N-1,0}^{*} \\ t_{N-1,1}^{*} \\ \vdots \\ t_{N-1,N-1}^{*} \end{bmatrix}$$

Since the columns of T^{-1} are independent, they form a basis for the N-dimensional space

And after this, I am showing the transformation here. So, this is the input data here and this is the T inverse I have the T inverse and this is a transform data F naught, F1 the capital F1 these are the transform data. Now, in this case, I can represent like this. Now, in this case, the columns of T inverse are independent. So, if I see the columns, these columns suppose these columns the columns of T inverse are independent and they form a basis for the N-dimensional space. So, this column is independent, this is independent, this is independent.

(Refer Time Slide: 21:56)



Examples of Unitary Transform

$$\mathbf{T} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & j \\ j & 1 \end{bmatrix}$$

$$\mathbf{T}^{-1} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -j \\ -j & 1 \end{bmatrix}$$

2D DFT

Other Examples : **DCT** (Discrete Cosine Transform)

DST (Discrete Sine Transform)

DHT (Discrete Hardmard Transform)

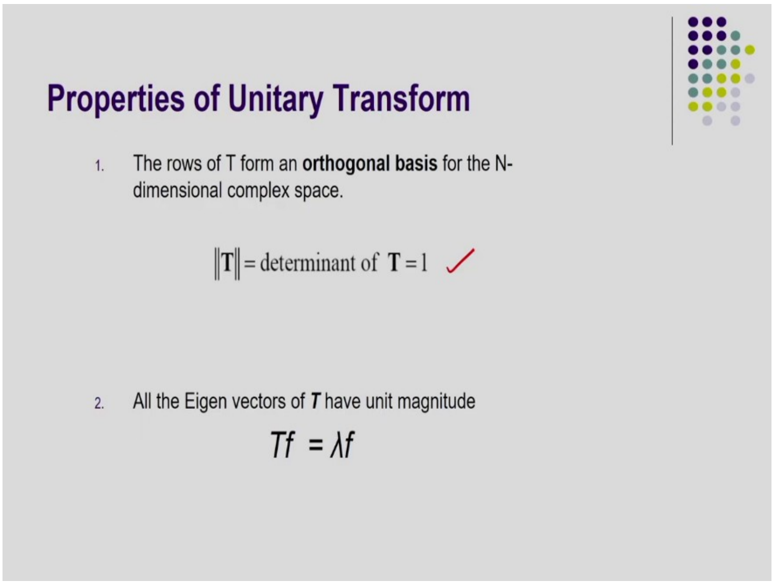
KLT (Karhunen Loeve Transform)

And I can give one example of the unitary transformation, here the 2D DFT I am considering. So, the transformation matrix is something like suppose 1 by root 2, 1,j,j,1, this is the transformation matrix. Then I can determine T inverse the T inverse will be 1 by root 2, 1 minus j, minus j, 1, so this is my T inverse. So, this is the definition of unitary transformation.

So, already I have explained, what are the examples of the unitary transformation. One is the DCT, one is the DST Discrete Sine Transformation, and one is the Hadamard transformation, and this KL transformation is also unitary. But in this case of the KL transformation, the transformation matrix or the transformation kernel depends on the statistics of the input data.

In the case of the DCT, the transformation kernel is fixed. But in the case of the KL transformation, the transformation kernel is not fix it depends on the statistics of the input data but it is orthogonal.

(Refer Time Slide: 23:02)



Properties of Unitary Transform

1. The rows of T form an **orthogonal basis** for the N-dimensional complex space.

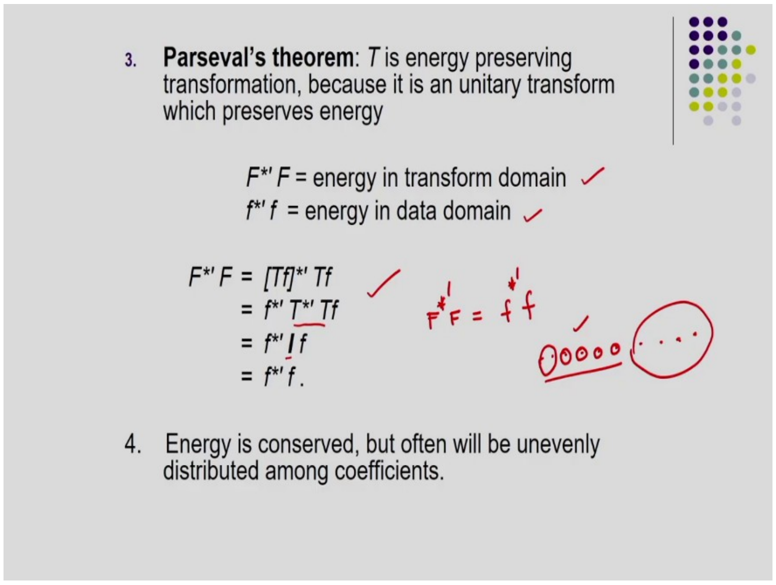
$$\|\mathbf{T}\| = \text{determinant of } \mathbf{T} = 1 \checkmark$$

2. All the Eigen vectors of ***T*** have unit magnitude

$$Tf = \lambda f$$

Now, the properties of the unitary transformation. The rows of the transformation matrix T form an orthogonal basis for the N-dimensional complex space that is very important property and one important thing is the determinant of T is equal to 1 that you can determine, the determinant of T is equal to 1. All the Eigenvectors of T have a unit magnitude that is another property, Tf is equal to lambda F by using this you can determine the Eigenvalues and Eigenvectors, so all the Eigenvectors of T have unit magnitude.

(Refer Time Slide: 23:39)



3. **Parseval's theorem**: *T* is energy preserving transformation, because it is an unitary transform which preserves energy

$F^{*\prime} F$ = energy in transform domain ✓
$f^{*\prime} f$ = energy in data domain ✓

$$F^{*\prime} F = [Tf]^{*\prime} Tf = f^{*\prime} \underline{T^{*\prime} T} f = f^{*\prime} \underline{I} f = f^{*\prime} f.$$

$F^{*\prime} F = f^{*\prime} f$

4. Energy is conserved, but often will be unevenly distributed among coefficients.

Another important property is the Parseval's theorem. That means, T is energy preserving, the transformation is energy preserving because it is a unitary transformation which preserves energy that is called the Parseval's theorem. So, in this mathematics I have shown here, first I am calculating the energy in the transform domain F complex conjugate transpose F that corresponds to energy in transform domain.

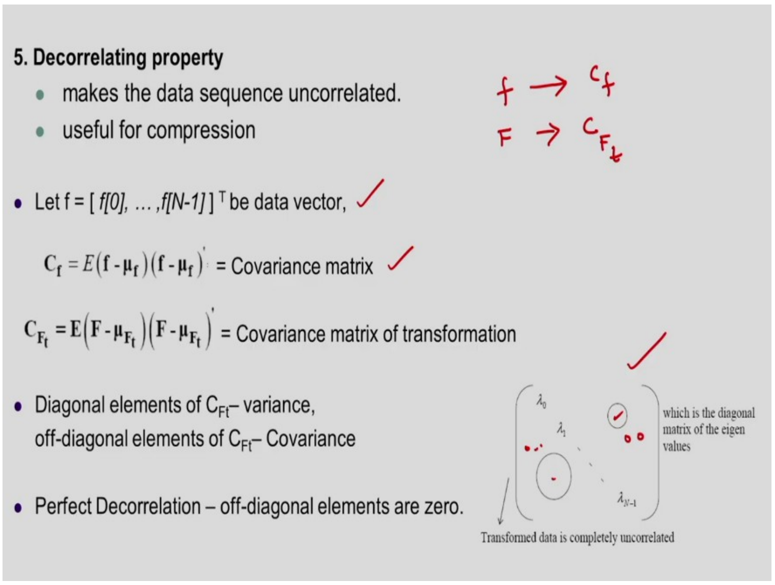
And another one is I can determine the energy in data domain that small f complex conjugate transpose f that is the energy in the Data Domain. After this, you can see these mathematics and if you see this T complex conjugate transpose and this part, this is nothing but the identity matrix, I is the identity matrix. So, from this, here you can see that F complex conjugate transpose F that is the energy into transform domain that is equal to the energy into data domain, so energy into data domain is this, so energy is preserved.

The energy is conserved, but one thing is that most of the energies are unevenly distributed among the coefficients. So, what is the meaning of this? Suppose the operator transformation, I am getting some coefficients, these are my coefficients but most of the energies are available in few coefficients. So, these energies are available like this and for the rest of the coefficients, the energy is very negligible.

That means, energy is conserved but will be unevenly distributed among the coefficients. So, most of the energies are available in these coefficients only, and remaining if you see the coefficient the energy is negligible. So, in case of the image compression, I can neglect these coefficients, because image compression is nothing but the compact representation of data.

So, I can neglect the redundant information. So, that is why I can consider only these coefficients, the coefficients having the significant energy I can consider and remaining coefficients I can neglect. So, this property is very important the energy is conserved, but most of the energies are available in a few coefficients.

(Refer Time Slide: 25:59)



5. Decorrelating property

- makes the data sequence uncorrelated.
- useful for compression

$f \rightarrow C_f$

$F \rightarrow C_{F_t}$

- Let f = [*f[0], … ,f[N-1]*] ^{T} be data vector, ✓

$C_f = E(f - \mu_f)(f - \mu_f)'$ = Covariance matrix ✓

$C_{F_t} = E(F - \mu_{F_t})(F - \mu_{F_t})'$ = Covariance matrix of transformation

✓

- Diagonal elements of C_{Ft} – variance,
  off-diagonal elements of C_{Ft} – Covariance
- Perfect Decorrelation – off-diagonal elements are zero.

$$\begin{pmatrix} \lambda_0 & & \\ & \lambda_1 & \\ & & \ddots & \\ & & & \lambda_{N-1} \end{pmatrix}$$

which is the diagonal matrix of the eigen values

Transformed data is completely uncorrelated

The next property is the Decorrelating property. So, in this property, you can see the input data is highly correlated, but after the transformation, the transform data will be less correlated or maybe you can consider uncorrelated. So, input data is highly correlated and after data transformation transform data will be uncorrelated that is the concept.

So, if I consider this is my data vector, corresponding to this data vector I can determine the covariance matrix and after the transformation, I am getting the transform data, the transformed data is F capital F, and my input device small f. So, for this transformed data also I can determine the covariance matrix, the covariance matrix is C F t and for this also I can determine the covariance matrix.

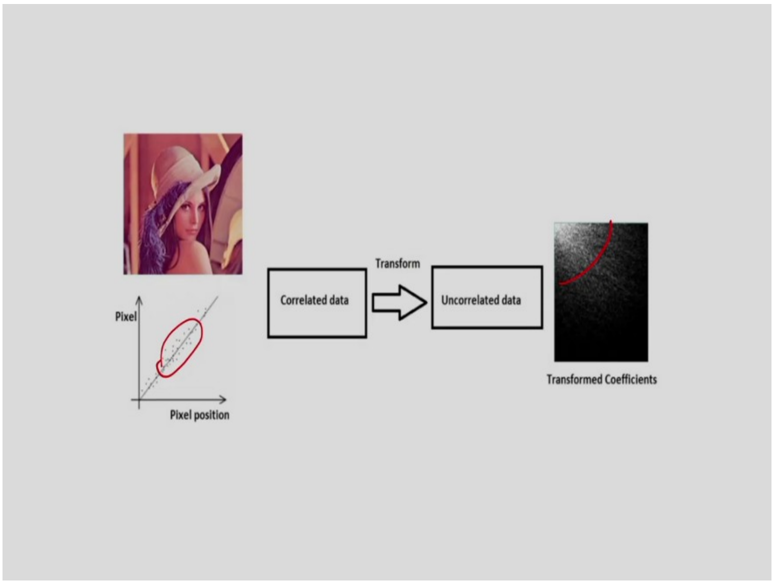
So, in this case, my original data is highly correlated, but after the transformation, I am having the covariance matrix something like this, this is the diagonal covariance matrix. The off-diagonal elements all these elements will be 0, if you see this 0, all these are 0. So, that means, the off-diagonal elements are 0 that corresponds to the perfect Decorrelating. So, that is the meaning of decorrelating property.

So, after the transformation that transforms data will be less correlated or maybe if I consider a perfect Decorrelation, then in this case the off-diagonal elements will be 0. So, these are the properties of the unitary transformation, so the first property is quite important, the rows of the transformation matrix they are independent and they form the basis for the N-dimensional space that is one first property and determinate of the transformation matrix T is equal to 1.

There is one important property, one is the Parseval’s theorem, the energy in the transform domain is equal to energy into data domain that is the very important property. Another property is energy complexion property. So, most of the energy is available in few coefficients. So, after the transformation, I am getting the transform coefficients and most of the energy is available in the few coefficients, so I can neglect the remaining coefficients for compact representation of data that is called energy complexion property.

And another property is the Decorrelating property. So, my input data is highly correlated and after the transformation, the transform data will be uncorrelated or maybe less correlated. So, if I have the diagonal covariance matrix that means, it is completely uncorrelated. So, these are the properties of the unitary transformation.

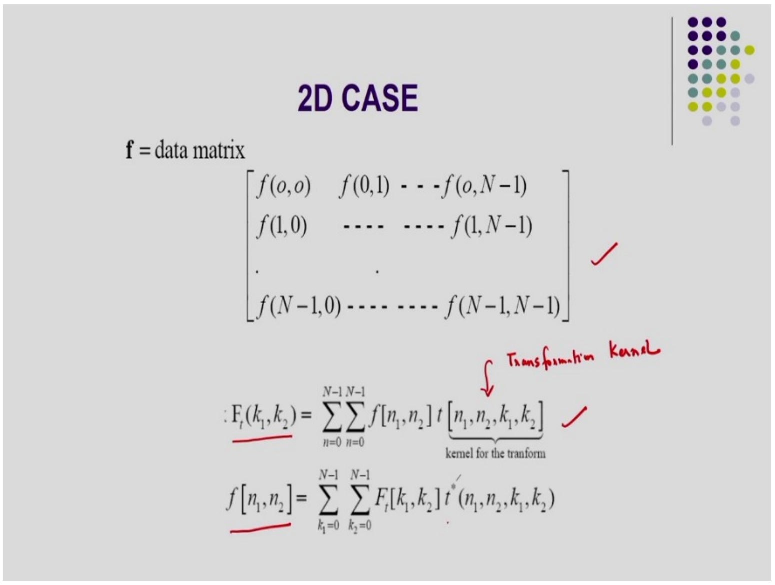
(Refer Time Slide: 28:40)



So, in this example, I have shown the first one is the image in the spatial domain. So, I have shown the pixel position and the pixels I am showing. So, you can see the correlated data, you can see the pixels if you see the pixels, so input is correlated. And after the transformation, I am getting the transform coefficient; you can see the transform coefficient here, so this portion is a transform corporation.

And, in this case, I am getting the uncorrelated data. So, that is the importance of the transformation. So, my input data is correlated, but after the transformation, I am getting the uncorrelated data.

(Refer Time Slide: 29:14)



2D CASE

f = data matrix

$$\begin{bmatrix} f(o,o) & f(0,1) & \cdots & f(o,N-1) \\ f(1,0) & \cdots & \cdots & f(1,N-1) \\ . & & . & \\ f(N-1,0) & \cdots & \cdots & f(N-1,N-1) \end{bmatrix} \checkmark$$

Transformation kernel

$$\underline{F_t(k_1,k_2)} = \sum_{n=0}^{N-1}\sum_{n=0}^{N-1} f[n_1,n_2]\, t \underbrace{[n_1,n_2,k_1,k_2]}_{\text{kernel for the tranform}} \checkmark$$

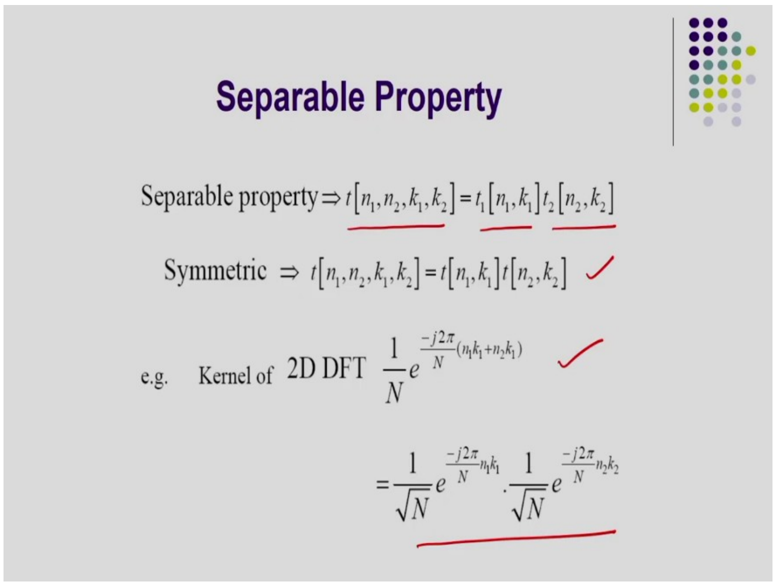
$$\underline{f[n_1,n_2]} = \sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1} F_t[k_1,k_2]\, t^{*}(n_1,n_2,k_1,k_2)$$

Now, let us consider the 2D transformation. So, one example I can give the 2D transformation mean in the image I can apply the 2D transformation. So, what is the 2D transformation here? So, first I am considering the data matrix f 00, f 01, f 0 N minus 1 this is one row, like this another row is f 10, and up to this f 1 N minus 1, so I am considering the data matrix.

Next one is I am considering the transformation. I am getting the transformed data, the transformed data is Ft k1 k2, there in this case the Ft the transform coefficients. And in this case, k1 and k2 are the row and the column indices in the transform array and I am considering the transformation kernel, the transformation kernel is t n1, n2, k1, and k2, so this is my transformation kernel.

And if I want to reconstruct the original data after the transformation, so in this case, you can see because I am considering the unitary transformation that means the t inverse is equal to the t complex conjugate transpose. So, I am considering this kernel, so I can reconstruct the original data.

(Refer Time Slide: 30:29)



Separable Property

Separable property $\Rightarrow \underline{t[n_1,n_2,k_1,k_2]} = \underline{t_1[n_1,k_1]}\,\underline{t_2[n_2,k_2]}$

Symmetric $\Rightarrow t[n_1,n_2,k_1,k_2] = t[n_1,k_1]\,t[n_2,k_2]$ ✓

e.g. Kernel of 2D DFT $\frac{1}{N}e^{\frac{-j2\pi}{N}(n_1k_1+n_2k_1)}$ ✓

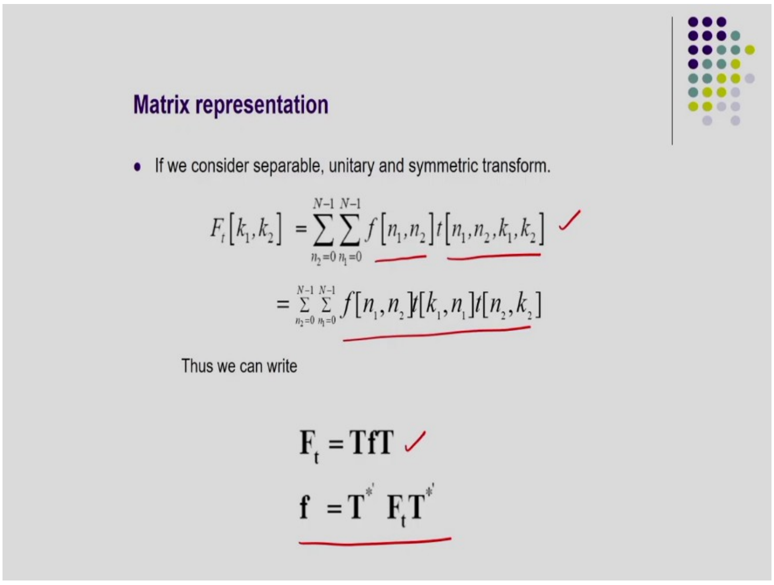
$$\underline{=\frac{1}{\sqrt{N}}e^{\frac{-j2\pi}{N}n_1k_1}\cdot\frac{1}{\sqrt{N}}e^{\frac{-j2\pi}{N}n_2k_2}}$$

Now, let us consider one property that is called the Separable property. So, already I have defined the transformation kernel, the transformation kernel is this, this is my transformation kernel and if I can do the separation like this the t1, the one kernel is t1 another kernel is t2. So, that means that there is transformation kernel t1, that is the kernel for the horizontal direction, and transformation kernel t2 that is for the vertical direction.

So, if I can separate like this, that is called the Separable property. One important point is separable transforms are easy to implement, why it is easy to implement? Because the transformation can first be applied along the rows and after this, I can apply the transformation along the columns. And in this case, if I consider the kernels along the rows and the columns, I have the identical function, then in this case I have the symmetric function, the symmetric property.

What is the meaning of the symmetric? If the kernels along the row and the columns have the identical function then this property is called a symmetric property. And if I considered a kernel product to the 2D DFT, so this is the kernel for the 2D DFT, that is separable you can see. So, based on the separable property, you can see, I can determine the 2D DFT by using the 1D DFT. So, I can apply 1D DFT along the rows and after this, I can apply that 1D DFT along with the columns because of this property the separable property.

(Refer Time Slide: 32:05)



Matrix representation

- If we consider separable, unitary and symmetric transform.

$$F_t[k_1,k_2] = \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \underline{f[n_1,n_2]}\,t\underline{[n_1,n_2,k_1,k_2]} \checkmark$$

$$= \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \underline{f[n_1,n_2]t[k_1,n_1]t[n_2,k_2]}$$

Thus we can write

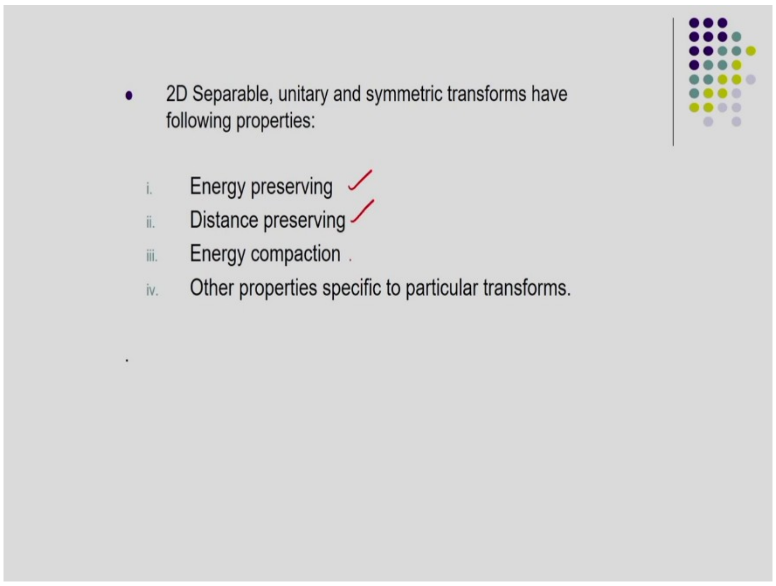
$$\mathbf{F_t = TfT} \checkmark$$

$$\underline{\mathbf{f = T^{*'}\ F_t T^{*'}}}$$

So, based on this separable and symmetric property and I am considering the unitary transformation. If you see, already I have defined this, this is the transformation and f n1 is the mind data input data, the 2D data and I am considering the transformation matrix these are transformation kernel and I am considering T is the transformation kernel and I am applying the separable property and the symmetric property.

So, if I apply the symmetric property and the separable property, I am having this one. So, I can write these in the matrix from like this Ft is equal to T f T, and what about the inverse transformation. This is my inverse transformation T complex conjugate transpose Ft T complex conjugate transpose that is my inverse transformation because I am considering the unitary transformation.

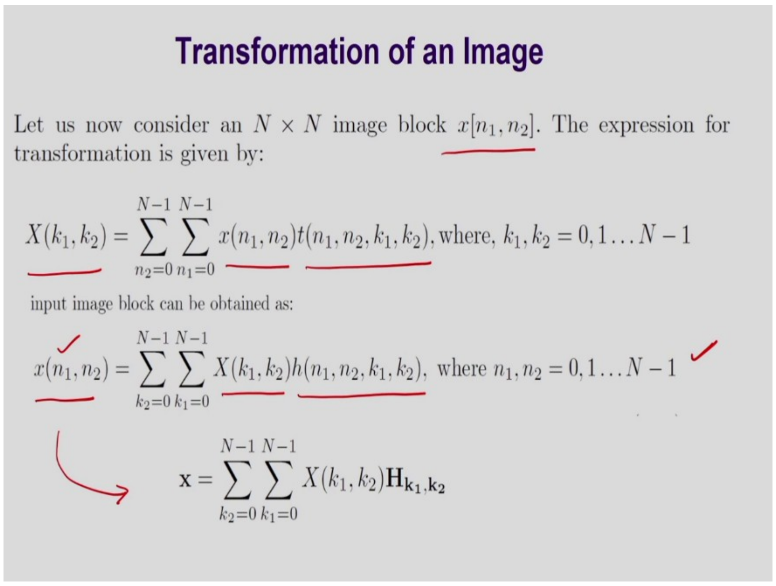
(Refer Time Slide: 32:57)



- 2D Separable, unitary and symmetric transforms have following properties:
  - i. Energy preserving ✓
  - ii. Distance preserving ✓
  - iii. Energy compaction .
  - iv. Other properties specific to particular transforms.

So, in case of the 2D transformation, 2D separable and I am considering the unitary transformation, and also I am considering the symmetric transformation. And this property again I am considering that this property already I have explained, one is the energy preserving property, the distance preserving property, energy compaction property and other properties like the decorrelating property, these properties are again applicable in the 2D transformation.

(Refer Time Slide: 33:26)



Transformation of an Image

Let us now consider an $N \times N$ image block $\underline{x[n_1, n_2]}$. The expression for transformation is given by:

$$\underline{X(k_1,k_2)} = \sum_{n_2=0}^{N-1}\sum_{n_1=0}^{N-1} \underline{x(n_1,n_2)}\,\underline{t(n_1,n_2,k_1,k_2)}, \text{where, } k_1, k_2 = 0, 1 \ldots N-1$$

input image block can be obtained as:

$$\underline{\overset{\checkmark}{x(n_1,n_2)}} = \sum_{k_2=0}^{N-1}\sum_{k_1=0}^{N-1} \underline{X(k_1,k_2)}\,\underline{h(n_1,n_2,k_1,k_2)}, \text{ where } n_1, n_2 = 0, 1 \ldots N-1 \checkmark$$

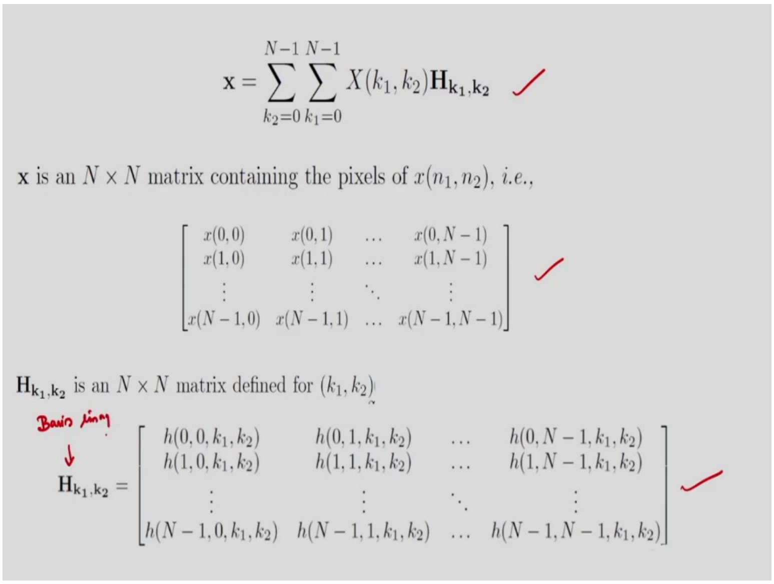
$$\mathbf{x} = \sum_{k_2=0}^{N-1}\sum_{k_1=0}^{N-1} X(k_1,k_2)\mathbf{H}_{\mathbf{k_1,k_2}}$$

And if I consider one image, image is nothing but the 2D array of numbers, then again I can show you the 2D transformation. So, I am considering one image block, the image block is x and the n1, n2, and that the size of the image is N cross N. So, for the transformation I am getting the transform data, the transform data X k1, k2.

And in this case, this is my input image and this is my transformation kernel, and what is my inverse transformation? Inverse transformation means, I am getting the input image from the transform coefficients. These are my transform coefficients X k1, k2 and this is my kernel for the inverse transformation.

So, this can be represented into matrix form, if you consider this equation there will be n square number of similar equations defined for each pixel element. So, that means, I have n square number of similar equations. So, that can be represented in the matrix form I can represent like this.

(Refer Time Slide: 34:28)



$$\mathbf{x} = \sum_{k_2=0}^{N-1} \sum_{k_1=0}^{N-1} X(k_1,k_2)\mathbf{H_{k_1,k_2}} \checkmark$$

$\mathbf{x}$ is an $N \times N$ matrix containing the pixels of $x(n_1,n_2)$, *i.e.*,

$$\begin{bmatrix} x(0,0) & x(0,1) & \dots & x(0,N-1) \\ x(1,0) & x(1,1) & \dots & x(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ x(N-1,0) & x(N-1,1) & \dots & x(N-1,N-1) \end{bmatrix} \checkmark$$

$\mathbf{H_{k_1,k_2}}$ is an $N \times N$ matrix defined for (k_1,k_2)

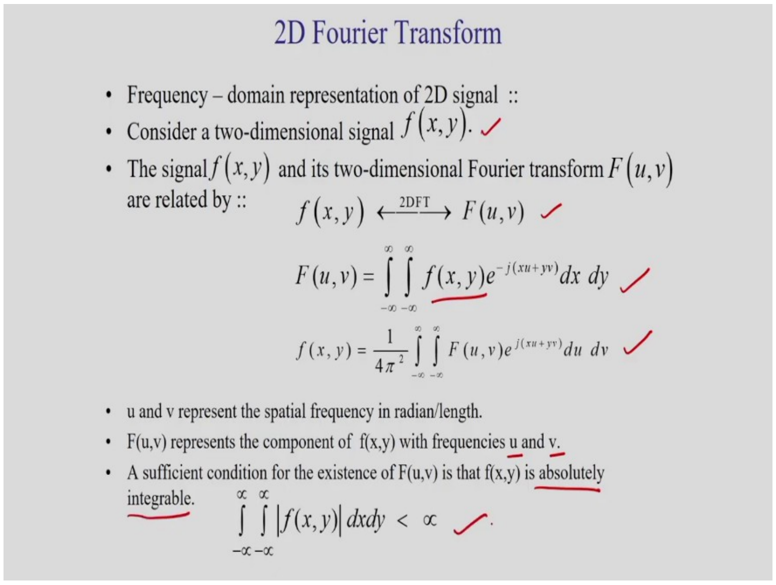
$$\overset{\text{Basis img}}{\downarrow}$$

$$\mathbf{H_{k_1,k_2}} = \begin{bmatrix} h(0,0,k_1,k_2) & h(0,1,k_1,k_2) & \dots & h(0,N-1,k_1,k_2) \\ h(1,0,k_1,k_2) & h(1,1,k_1,k_2) & \dots & h(1,N-1,k_1,k_2) \\ \vdots & \vdots & \ddots & \vdots \\ h(N-1,0,k_1,k_2) & h(N-1,1,k_1,k_2) & \dots & h(N-1,N-1,k_1,k_2) \end{bmatrix} \checkmark$$

So, again I am writing this one and what is my input image? The input image are like this. These are mainly the pixels, pixels of x n1, n2. So, I am writing this one and this is my matrix, the matrix is H k1, k2, so that is my matrix. So, this H k1, k2 is an N-by-N matrix defined for the variables k1 and k2. So, the image block x can be represented by a weighted summation of n square images, so I am repeating this.

So, a particular image block, x can be represented by a weighted summation of n square images and each of the size N cross N and the width of these linear combinations are the transform coefficients capital X k1, k2 and this H k1 matrix, this matrix is called is to the basis image.

(Refer Time Slide: 35:39)



2D Fourier Transform

- Frequency – domain representation of 2D signal ::
- Consider a two-dimensional signal $f(x,y)$. ✓
- The signal $f(x,y)$ and its two-dimensional Fourier transform $F(u,v)$ are related by ::

$$f(x,y) \xleftrightarrow{\text{2DFT}} F(u,v)$$

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \underline{f(x,y)} e^{-j(xu+yv)} dx\, dy$$

$$f(x,y) = \frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v) e^{j(xu+yv)} du\, dv$$

- u and v represent the spatial frequency in radian/length.
- F(u,v) represents the component of f(x,y) with frequencies u and v.
- A sufficient condition for the existence of F(u,v) is that f(x,y) is absolutely integrable.

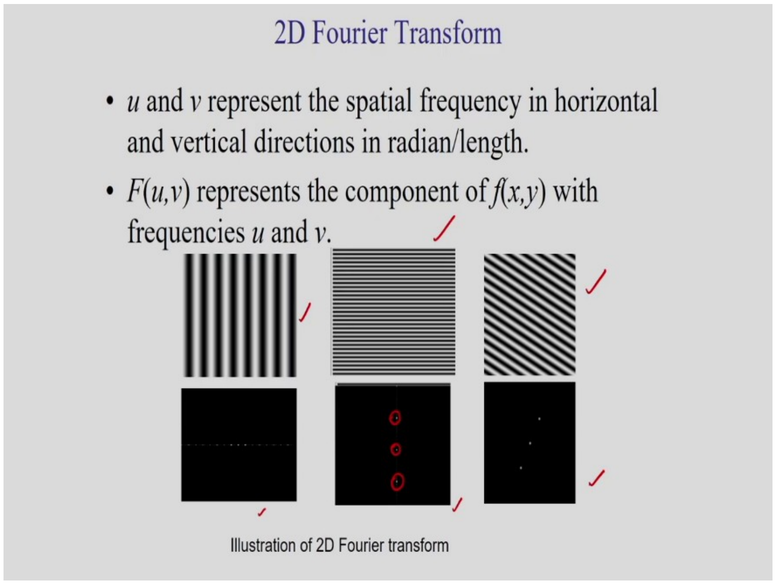
$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |f(x,y)|\, dxdy < \infty$$

Now, let us consider the 2D Fourier transform. So, how to define the 2D Fourier transform, here you can see. So, I am considering my input image f x y and this is my DFT pair. So, f x y, If I take the 2D DFT then I will be getting F u v. The F u v means the Fourier transform of the input image, the input image is f x y and this is my DFT formula that is in the continuous domain F u v, I can get from f x y e to the power minus j xu plus yv dx dy and this is my inverse Fourier transformation.

Now, in this case, you can see u and v, u and v is nothing but a spatial frequency in radian per length. What is u? u is the spatial frequency in the x-direction and v is the spatial frequency along the y-direction. So, in case of the f u v, I am considering two frequencies one is u another one is v the spatial frequency, and but a Fourier transform I think already you know this condition that f x y should be absolutely integrable.

So, this condition is important for the Fourier transformation. Now, if you see this example, I am determining the Fourier transform of the images, I am considering three images and you can see what will be the Fourier transform of these images. So, first images you can see.

(Refer Time Slide: 37:12)



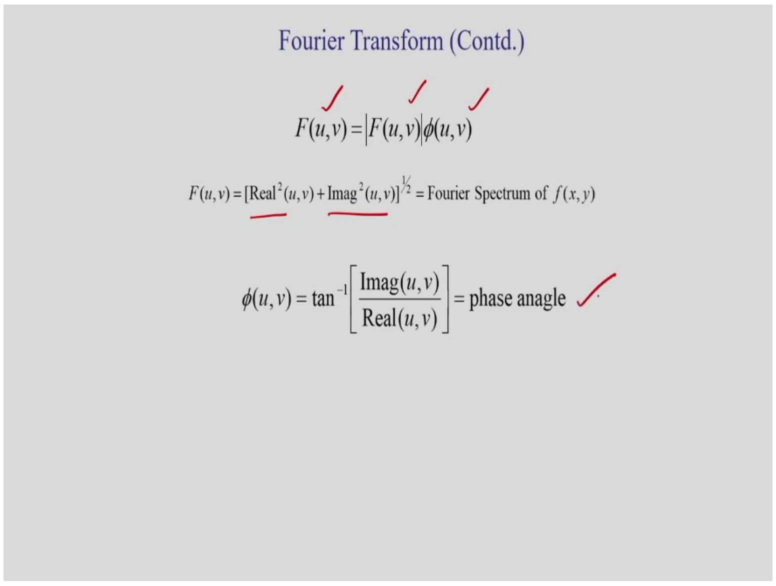
2D Fourier Transform

- u and v represent the spatial frequency in horizontal and vertical directions in radian/length.
- $F(u,v)$ represents the component of $f(x,y)$ with frequencies u and v.

Illustration of 2D Fourier transform

That is the first image, corresponding to the first image I have the Fourier transform that I am getting the spatial frequencies, corresponding to the second image also, I have the horizontal lines then, in this case, I have the Fourier transform you can see these points here. And for that this image also the third image I have the Fourier transform that is a 2D Fourier transform.

(Refer Time Slide: 37:35)



Fourier Transform (Contd.)

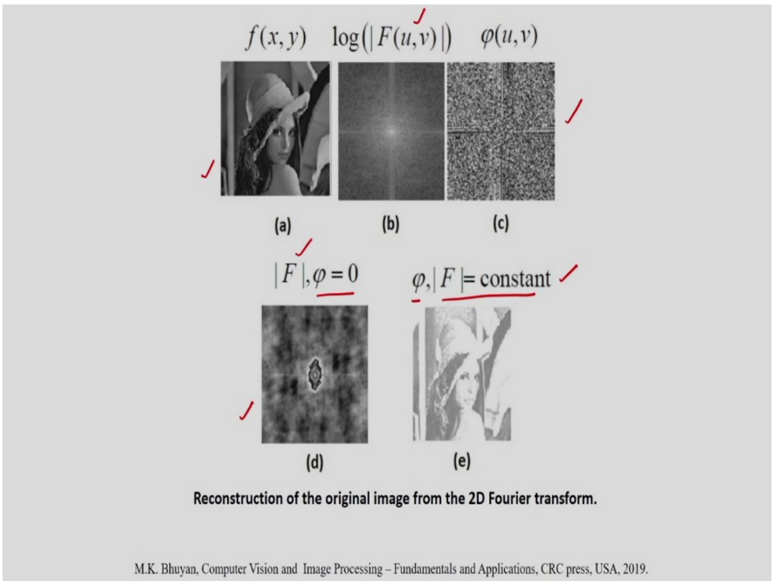
$$F(u,v) = |F(u,v)|\phi(u,v)$$

$$F(u,v) = [\text{Real}^2(u,v) + \text{Imag}^2(u,v)]^{1/2} = \text{Fourier Spectrum of } f(x,y)$$

$$\phi(u,v) = \tan^{-1}\left[\frac{\text{Imag}(u,v)}{\text{Real}(u,v)}\right] = \text{phase anagle}$$

This Fourier transform can be represented in the polar form, that F u v is the Fourier transform that can be represented in the polar form. So, I have the magnitude part and also the phase angle. So, I have two information one is the magnitude information and the other one is the phase information. And in case of the F u v, it has two components one is the real part another one the imaginary part, that is the Fourier spectrum of the input signal, the input signal is f x y and by using this formula I can determine the phase angle.

(Refer Time Slide: 38:08)



Reconstruction of the original image from the 2D Fourier transform.

M.K. Bhuyan, Computer Vision and Image Processing – Fundamentals and Applications, CRC press, USA, 2019.

Now, in this example, I want to show what is the meaning of the phase information and what is the meaning of the magnitude information. Phase information represents the edge information or the boundary information of the objects that are present in the image and for applications like medical image analysis, this phase information is very important. And what is the meaning of the magnitude?

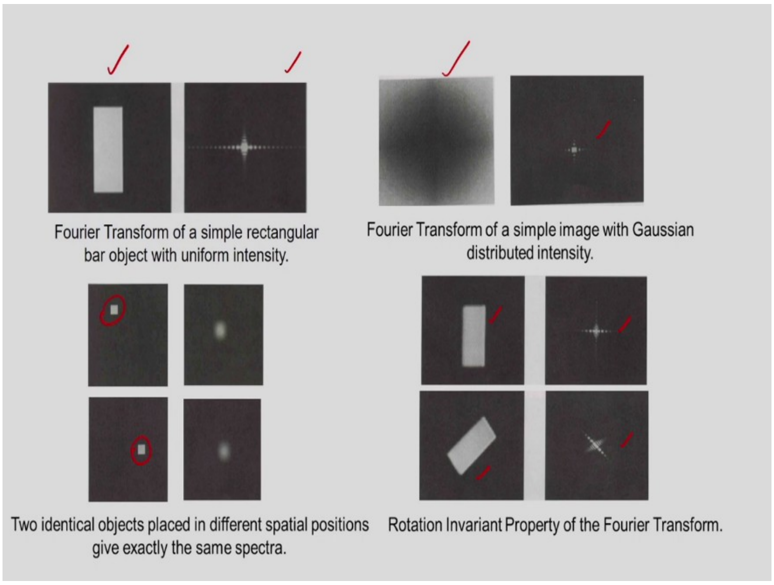
A magnitude tells how much of a certain frequency component is present in the image. And phase information tells where the frequency component is present in the image. Corresponding to this example, if you see I have the input image f x y and I have two plot, one plot is the magnitude information I am plotting, so first is the magnitude plot and second one is the phase information I am plotting.

So, in case of the magnitude plot, I am using the log transformation, I will explain what is the log transformation. Log transformation is used to compress the dynamic range of an image for better visualization. So, you have seen here I have two information one is the magnitude information, another one is the phase angle information. And in this case, you can see I am doing the reconstruction of the original image from the 2D Fourier transform.

So, in the first case, I am considering the magnitude information, but I am not considering the phase angle information, phase information I am not considering. And this is my reconstructed image that means, you cannot reconstruct the image only by considering the magnitude information. In the second case, I am considering the phase information, but I am not considering the magnitude information, magnitude is constant then also this is the reconstructed image.

So, in this example, you can see that I want to reconstruct the original image by considering the magnitude information and the phase angle information. If I only consider the magnitude information or if I only considered phase information, the perfect reconstruction is not possible. So, for perfect reconstruction, I need both magnitude information and the phase information.

(Refer Time Slide: 40:14)


Fourier Transform of a simple rectangular bar object with uniform intensity.
Fourier Transform of a simple image with Gaussian distributed intensity.
Two identical objects placed in different spatial positions give exactly the same spectra.
Rotation Invariant Property of the Fourier Transform.

And in this example, I have shown the Fourier transform of a simple rectangular bar. So, my input image this and corresponding to this I have the Fourier transform of this, because for a rectangular function if I take the Fourier transform, the Fourier transform will be that sync function. The second case I am considering the Fourier transform of a simple image with Gaussian distributed intensity.

So, if you see the intensity of this one, this is Gaussian distributed intensity, the Fourier transform of a Gaussian function is Gaussian. So, that means, in this case also I am getting that Gaussian distribution here, the third example I am considering two identical objects placed in different spatial position and corresponding to this if you see this here, this object is placed here, this object is placed here, corresponding to this I am getting the exactly the same Fourier spectrum, the Fourier spectrum I am getting.

And in our last example, I am considering the rotation invariant property of the Fourier transform. So, this object is rotated, if you see, then in this case, if it is rotated, the Fourier transform is also rotated that is called the rotational invariant property of the Fourier transform.

So, up till now, I discussed about the concept of the Fourier transform. The Fourier transform is quite important to see the frequency component present in the signal. So, in an image, I have the high-frequency information and another one is the lower frequency information. So, in my next class, I will discuss another important information that is the Discrete Cosine Transformation.

And also, I may discuss the fundamental concept of the KL transformation. In case of Fourier Transform and in case of the DCT, the transformation kernel is fixed. But in case of the KL transformation, the transformation kernel is not fixed; it depends on the statistics of the input data. So, next class, I will discuss about the DCT and the KL transformation. So, let me stop here today. Thank you.