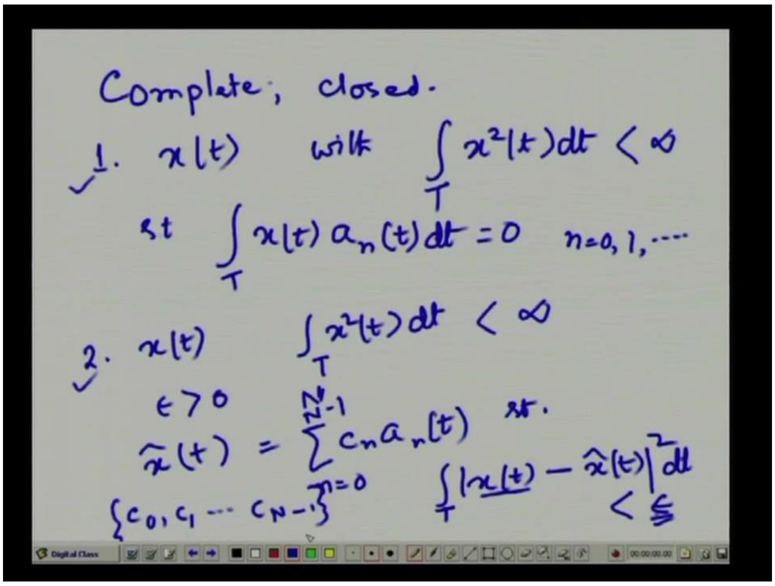
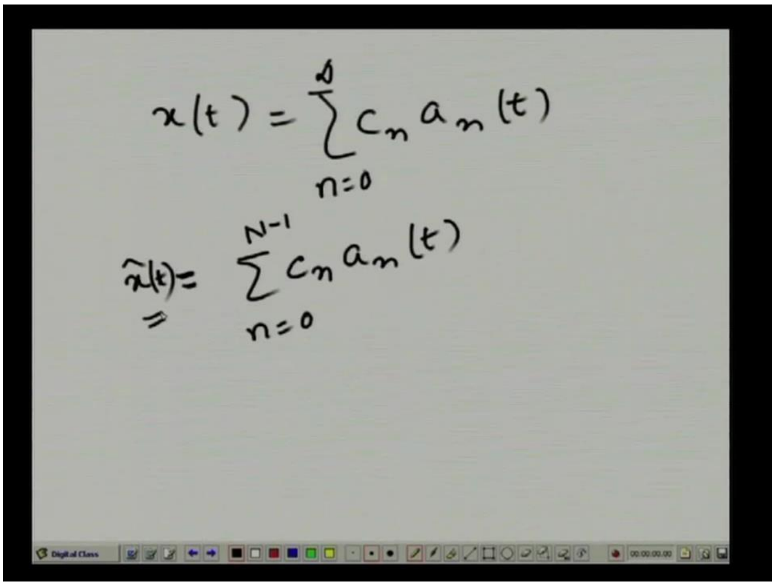
Digital Image Processing.
Professor P. K. Biswas.
Department of Electronics and Electrical Communication Engineering.
Indian Institute of Technology, Kharagpur.
Lecture-21.
Image Transformation-2.

(Refer Slide Time: 0:37)



Welcome to the course on digital image processing. Then using this complete set of orthogonal functions we can go for a finite expansion of a signal x(t) using the finite number of expansion coefficients C0, C1, upto CN-1, as is shown here. So I have a finite set of expansion coefficients. So from this discussion what we have seen is that an arbitrary continuous signal x(t) can be represented by the series summation of a set of orthogonal basis functions.

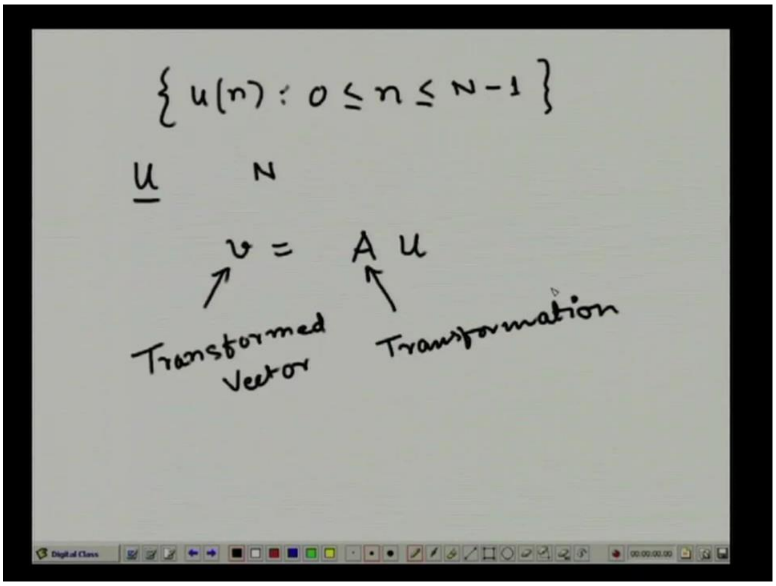
(Refer Slide Time: 1:06)



And this series expansion is given as x(t) is equal to Cn an(t), where n varies from 0 to infinity, if I go for infinite expansion or this can also be represented as we have seen by finite expansion finite series expansion in this case this will be represented by Cnan(t) where n will now vary from 0 to N capital N-1. So this is x hat (t), so obviously we are going for an approximate representation of x(t) not a complete expansion not the exact representation of x(t). So this is the case that we have for continuous signals x(t).

But, in our case we are not dealing with the continuous signal but we are dealing with the discrete signals. So in case of discrete signals what we have is a set of samples or a series of samples.

(Refer Slide Time: 2:30)



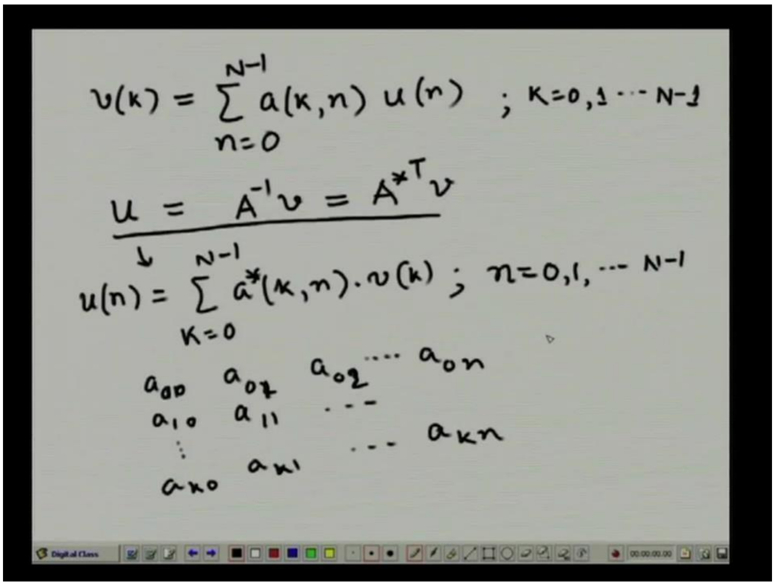
So the series of samples can be represented by say u(n) where 0 less than or equal to n less than or equal to capital N-1. So we have a series of discrete samples u(n), so in this case we have capital N number of samples.

So obviously you can see that this is a one dimensional sequence of samples. And because it is one dimensional sequence of samples and the sample size is capital N, that that is we have capital N number of samples. So I can represent this set of samples by a vector say u of dimension capital N. So I am representing this by a vector u of dimension capital N. And for transformation what I do is I pre multiply this vector u by a unitary matrix A, of dimension n by n.

So given this vector u, if I pre multiply this with a unitary matrix capital A, where the dimension of this unitary matrix is n by n. So you find that this u is a vector of dimension n. And I have a matrix, a unitary matrix of dimension n by n. So this multiplication results in another vector v. So this vector v we call as a transformed vector or transformation vector. This is transformed vector and this unitary matrix A is called the transformation matrix.

So what I have done is I have taken an n dimensional vector u pre multiplied by pre multiplied that n mi done n dimensional vector u by a unitary matrix of dimension n by n. So after multiplication I got again an n di n dimensional vector v. Now so by matrix equation this is v equal to A times u.

(Refer Slide Time: 5:02)



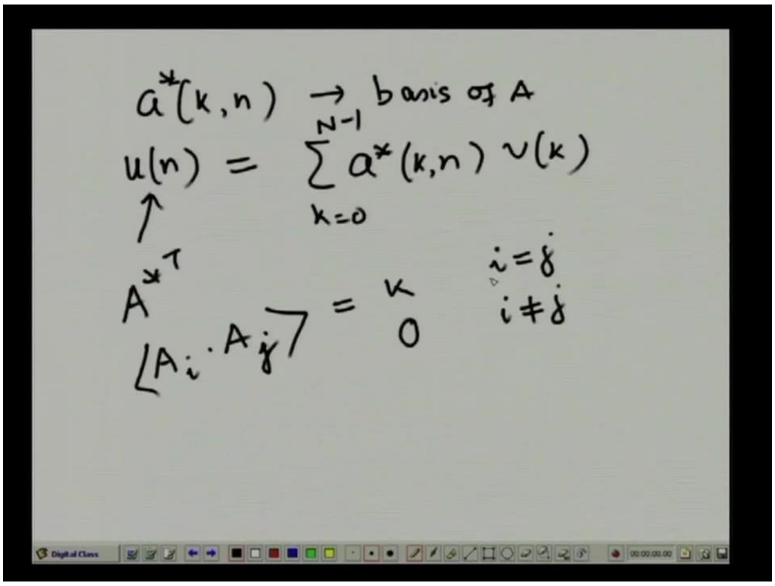
If I expand this, so now what I do is, I expand this matrix equation. So if I expand this matrix equation this can be represented as a series summation which will be given by v(k) is equal to a(k,n) into u(n), where n varies from 0 to capital N-1.

And this has to be computed for k equal to 0, 1, upto N-1. So I get the all the n elements of the vector v(k). Now if A is a unitary matrix then from vector v I can also get back our original vector u, so for doing that what we will do is we will pre multiply v by A inverse. So this should give me the original vector u, and this A inverse v because this is an unitary matrix will be nothing but A conjugate transpose v.

And if I represent this same equation in the form of a series summation this will come out to be u(n) is equal to a*(k,n) times v(k), where k will now vary from 0 to N-1. And this has to be computed for all values of n varying from 0,1, upto N-1. Now you find that what is this a*(k,n), now if I represent this a(k,n) or if I expand this matix a(k,n) this is of the form a11 or a01, a02, a03 like this a0n, a10, sorry this is a00, a01, a02 upto a0n. This will be a10, a11, so it will go like this and finally I will have ak0, ak1, like this I will have ak,n.

Now find that in this expression we are multiplying a(k,n) by v(k), a(k,n)* which is the conjugate of a(k,n) into v(k). Now this a(k,n)* is nothing but the column vector of matrix a*. So if I have this matrix a, this a(k,n)* is nothing but a column vector of matrix A*. So these column vectors, or column vectors of matrix A* transpose.

(Refer Slide Time: 8:29)



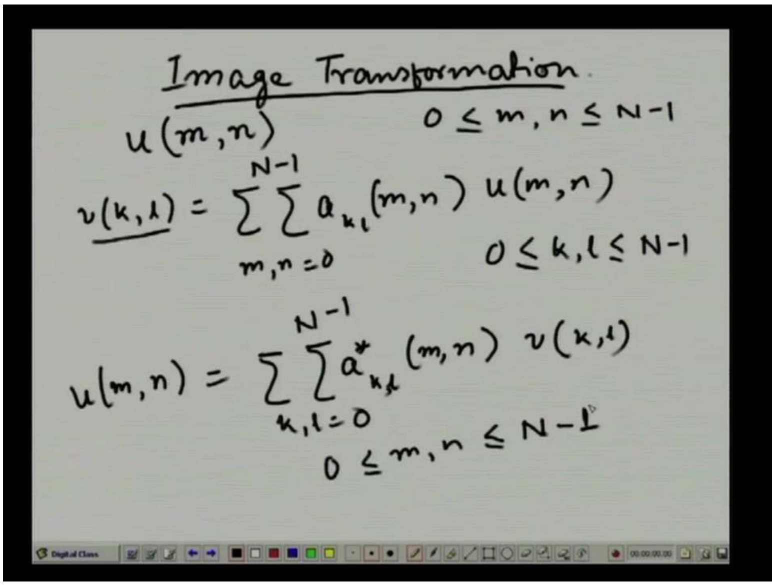
So these column vectors or a(k,n)* a(k,n)*, this columns vectors are actually called the basis vectors of the matrix A. And you do remember this matrix A is an unitary matrix. And here what we have done is the sequence of samples u(n) or vector u(n) has been represented because we have represented a(n), u(n) as summation of a*(k,n) into v(k) for k equal to 0 to N-1. So this vector u(n) has been represented as a series summation of a set of basis vectors. Now if this basis vector have to be orthogonal or orthonormal, then what is the property it has to follow.

So if we have a set basis vectors and in this case we have said that the columns of A* transpose this forms the set of basis vectors. So if I take any two columns and take the dot product of those two those two columns the dot product is going to be non 0 the dot product is going to be 0. And if I take the dot product of the column with itself these dot product is going to be non 0.

So if I take a column say A column i and take the dot product of Ai with Ai, or I take two columns Ai and Aj and take the dot product of these two columns. So these dot products will be equal to some constant k, whenever i is equal to j. And this will be equal to 0 whenever i is not equal to j. So if this property is followed then the matrix A will be an unitary matrix.

So in this case we have represented the vector v or vector u by a series summation of a set of basis vectors. So this is what we have got in case of one dimensional signal or an one dimensional vector u.

(Refer Slide Time: 11:02)



Now we are talking about image transformation, so in our case our interest is on image transformations. Now the same concept of representing a vector as a series summation of a set of basis vectors, can also be extended in case of an image.

So in case of an image, the vector u that we have defined in the earlier case, now it will be a two dimensional matrix. So u instead of being a vector now it will be a two dimensional matrix and we represent this by u(m,n) where m and n are row and column indices, where 0 is less than or equal to m,n and less than or equal to N-1.

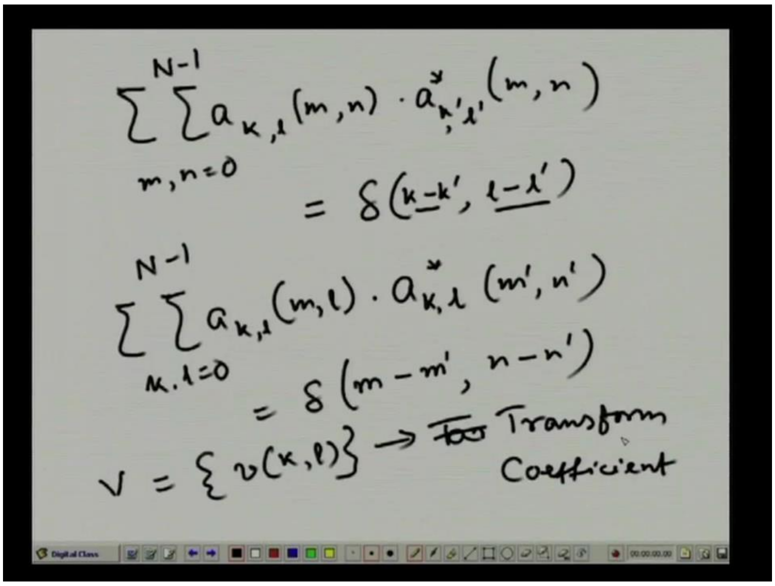
So see that we are considering an image of dimension capital N by capital M. Now transformation on this image can be represented as v(k,l) will be equal to again we take the series summation akl(m,n) into u(m,n) where m and n vary from 0 to capital N-1, ok. So here you find that akl is a matrix again of dimension capital M by capital N. But in this case the matrix is itself has an index kl and this computation v(k,l) has to be done for 0 less than or equal to k,l less than or equal to N-1.

So this clearly shows that the matrix that we are taking this is of dimension capital M by capital N and not only that we have capital N into capital N that is N square, capital N square number of such matrices or such unitary matrices. So this akl(m,n) because kl, k and l both of them take the values from 0 to capital N-1, so I have capital N square number of unitary matrices.

And from this vkl, which is in this case the transformation matrix I can get back this original matrix u(m,n) by applying the inverse transformation. So in this case u(m,n) will be equal to again double summation a*k dash, l dash, into v sorry a*k,l into let me rewrite this, we will have, u(m,n) will be given by double summation a*k,l(m,n) v(k,l) where k,l varies from 0 to N-1. And this has to be computed for 0 less than or equal to m,n less than or equal to N-1.

So we find that by extending the concept of series expansion of one dimensional vectors to two dimension we can represent an image as a series summation of basis unitary matrices. So in this case all of ak,l or a* all of ak,l(m,n) will be the unitary matrices. Now what is the orthogonality property what is meant by orthogonality property in case of the matrix.

(Refer Slide Time: 15:45)

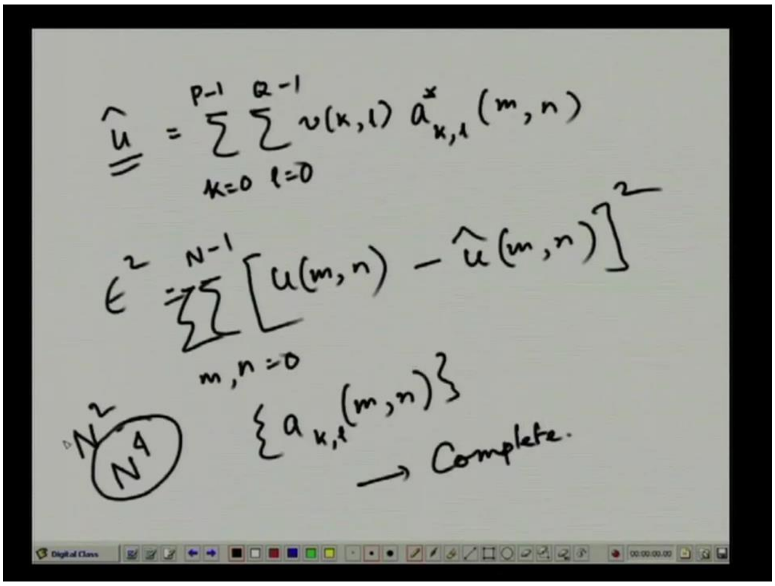


The orthogonality property says that for this matrix A it says that ak,l(m,n) into ak dash, l dash (m,n), if I take this summation for m,n equal to 0 to capital N - 1. This will be equal to a conical delta function of (k - k dash and l - l dash). So it says that this functional value will be equal to 1, whenever k is equal to k dash and l equal to l dash. In all other cases this summation will be 0 and the completeness says that if I take the summation ak,l(m,n) into ak,l sorry this should be ak dash l dash*, so ak,l*(m dash, n dash) summation is taken over k and l equal to 0 capital N - 1, this will be equal to conical delta function (m -m dash and n -n dash).

So it says that this summation will be equal to 1 whenever m is equal to m dash and n is equal to n dash. So the matrix by applying this kind of transformation the matrix v which we get which is nothing but set of v(k,l), this is what is the transform matrix or the transformation coefficients. So this is also called the transformed coefficients. So we find that in this particular case any arbitrary image is represented by a series summation of a set of basis images or set of unitary matrices.

Now if we truncate the summation. So in this case what we get is we get the set of coefficient and the coefficient size is same as the original image size, that is if we have n by n image or coefficient matrix will also be of n by n. Now is while doing the inverse transformation if I do not consider all the coefficient matrices, I consider a subset of it, in that case what we are going to get is an approximate reconstructed image.

(Refer Slide Time: 19:01)



And it can be shown that this approximate reconstructed image will enca will have an error a limited error if the basis matrices that we are considering the set of basis matrices or set of basis images that is complete. So this error will be minimized if the basis images that we consider is complete basis images. So in that case what we will have is the reconstructed image that u hat will be given by double summation say v(k,l) into a*k,l(m,n).

Now suppose l will vary from 0 to Q - 1 and say k will vary from 0 to P - 1. So instead of considering both k and l, varying from 0 to N - 1, I am considering only Q number of coefficients along l and P number of coefficients along k. So the number of coefficients that I am considering for reconstructing the image of a inverse transformation is P into Q instead of N square.

So this P into Q is using this P into Q number of coefficients I get the reconstructed image u hat, so obviously this u hat is not the exact image it is an approximate image because I did not consider all the coefficient values and the sum of squared error in this will be given by epsilon square equal to u(m,n) that is the original image minus u hat (m,n) which is the approximate reconstructed image square of this and you take the summation over m,n varying from 0 to N - 1.

And it can be shown that this error will be minimized if a set of basis images that is ak,l(m,n) this is complete. Now another point that is to be noted here if you compute the amount of computation that is involved you find that if N square is the image size. The number of computations or amount of computations that will be needed both for forward transformation and for inverse transformation will be of order N to the power 4.

So for doing this we have to have, we have to enquire tremendous amount of computation. So one of the problem is how to reduce this computational requirement when you go for inverse transformation or whenever we go for forward transformation, thank you.