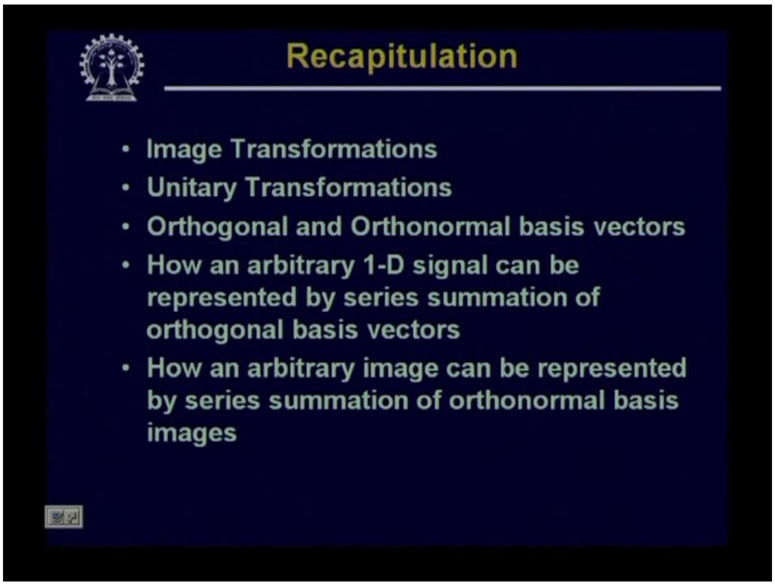
Digital Image Processing.
Professor P. K. Biswas.
Department of Electronics and Electrical Communication Engineering.
Indian Institute of Technology, Kharagpur.
Lecture-22.
Separable Transformation.

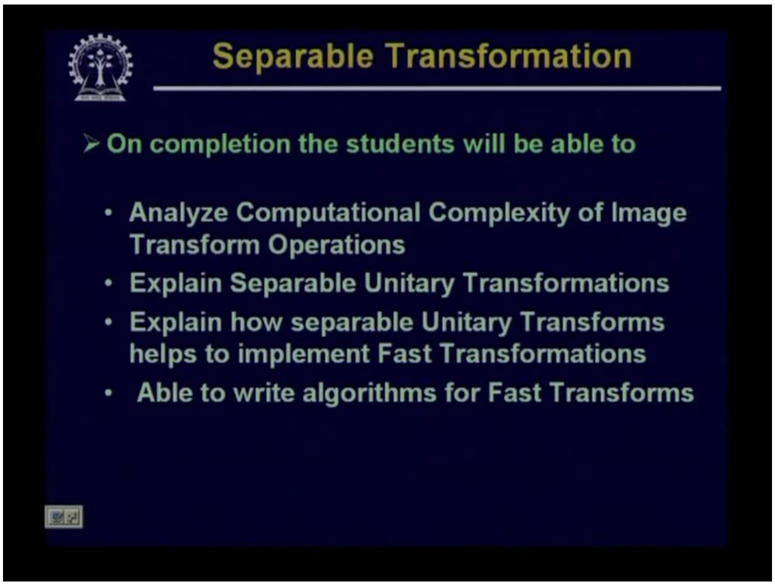
(Refer Slide Time: 0:40)



Hello, welcome to the video lecture series on digital image processing. So let us see what we have done in our last lecture, in our introductory lecture on image transformations we have said the basics of image transformation. We have seen what is meant by an unitary transform, we have also seen what is orthogonal and orthonormal basis vectors. We have seen how an arbitrary one dimensional signal can be represented by series summation of orthogonal basis vectors.

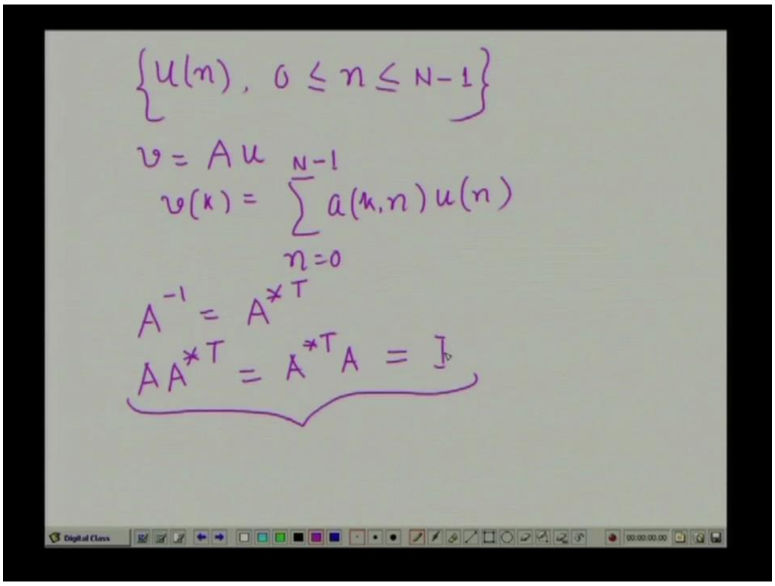
And we have also seen how an arbitrary image can be represented by series summation of orthonormal basis images. So when we talk about the image transformation basically the image is represented as a series summation of orthonormal basis images.

(Refer Slide Time: 1:30)



After today's lecture the students will be able to analyze the computational complexity of image transform operations. They will be able to explain what is meant by a separable unitary transformation. They will also know how separable unitary transforms help to implement fast transformations and ofcourse they will be able to write algorithms for fast transforms. So first let us see that what we have done in the last class. In the last class we have taken one dimensional sequence of the discrete signal samples, say given in the form u(n), where n varies from 0 to some capital N-1.

(Refer Slide Time: 2:31)



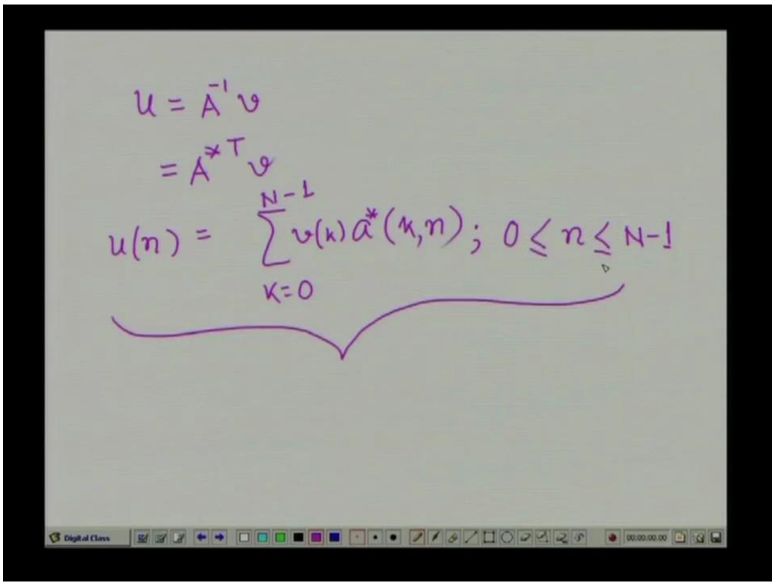
So we have taken initially a one dimensional sequence of discrete samples like this that is u(n) and we have found out what is meant by unitary transformation of this one dimensional discrete sequence. So by unitary transformation the unitary transformation of this one dimensional discrete sequence is given by say v is equal to A times u, where A is an unitary matrix. And this can be represented expanded in the form v(k) is equal to we have a(k,n)u(n), where n varies from 0 to capital N-1.

Assuming that we have capital N number of samples in the input discrete sequence. Now we say that this transformation is an unitary transformation, if the matrix A is an unitary matrix. So what is meant by an unitary matrix, the matrix A will be said to be a unitary matrix if it obeys the relation that A inverse, inverse of matrix A will be given by A conjugate transpose. That is if you take the conjugate of every element of matrix A and then the take then take the transpose of those conjugate elements then that should be equal to the inverse of matrix A itself.

So this says that A into A conjugate transpose that should be same as A conjugate transpose A which will be same as an identity matrix. So if this relation is true for the matrix A, then we say that A is an unitary matrix and the transformation which is given by this unitary matrix is a unitary transformation. So using this matrix A we go for unitary matrix, unitary transformation.

Now, once we have this transformation and we get the transformation coefficient say vk, or the transformed vector transformed sequerce sequence v. We should be also able to find out that how from this transformation coefficients we get back the original sequence u(n).

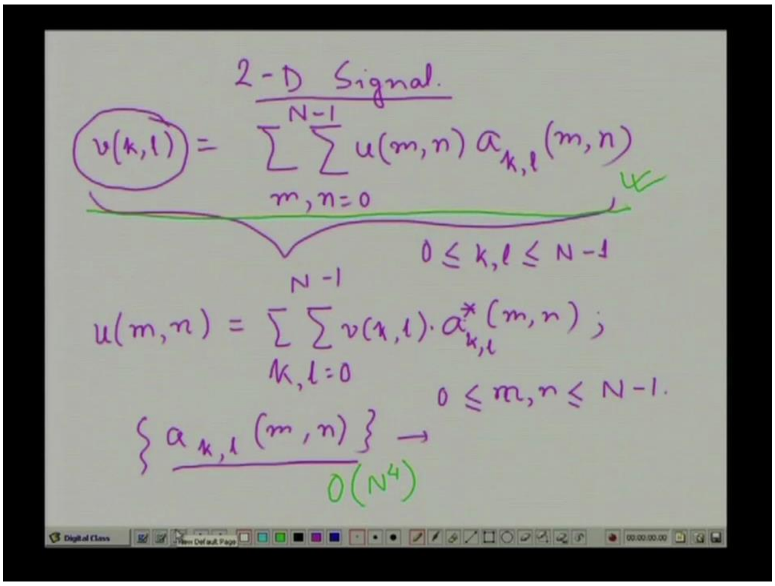
(Refer Slide Time: 5:41)



So this original sequence is obtained by a similar such relation which is given by u is equal to A obvi obviously it should be equal to A inverse v and in our case since A inverse is same as A conjugate transpose. So this can be written as A conjugate transpose v, and this expression can be expanded as u (n) is equal to summation v(k)a conjugate (k,n) where k varies from 0 to N-1. And we have to compute this for all values of n varying from 0 to N-1, so 0 less than or equal to n less than or equal to capital N-1. So by using the unitary transformation we can get the coefficients the transformation coefficients and using the inverse transformation we can obtain the input sequence input discrete sequence from the coefficient from this sequence of coefficients.

And this expression says that the input sequence u(n) is now represented in the form of a (**se**) series summation of a set of vectors or orthonormal basis vectors. So this is what we get in case of one dimensional sequence. Now let us see what will be the case in case of a two dimensional sequence.

(Refer Slide Time: 7:33)



So for a two dimensional sequence, see if I go for the case of two dimensional signals, then the same transformation equations will be of the form v(k,l) is equal to we have to have double summation u(m,n) into ak,l(m,n), where both m and n varies from 0 to capital N-1.

So here u(m,n) is the input image it is the two dimensional image, again we are transforming this using the unitary matrix A and in the expanded form the expression can be written like this v(k,l) is equal to double summation u(m,n)ak,l(m,n), where both m and n varies from 0 to infinity. And this has to be computed for all the values of k and l where k and l varies from 0 to N-1. So all k and l will be in the range 0 to N-1.

In the same manner we can have the inverse transformation so that we can get the original two dimensional matrix from the transformation coefficient matrix and this inverse transformation in the expanded form can again be written like this. So from v(k,l) we have to get back u(m,n) so you can write it as u(m,n) again is equal to double summation v(k,l) into a*k,l(m,n), where both k and l will vary in the range 0 to capital N-1. And this we have to compute for all values of m and n in the range 0 to capital N-1.

Where this image transform that is ak,l(m,n), this is nothing but a set of complete orthonomal normal discrete basis functions. So this ak,l(m,n) this is a set of complete orthonormal basis functions. And in our last class we have said what is meant by the complete set of orthonormal basis functions. And in this case this quantity, the v(k,l) what we are getting these are known as transformed coefficients.

Now let us see that what will be the computational complexity of these expressions. If you take any of this expressions, say for example the forward transformation where we have this particular expression v(k,l) is equal to double summation u(m,n)ak,l(m,n) where m and n vary from 0 to capital N-1, that means both m and n, m will vary from 0 to capital N-1, n will also vary from 0 to capital N-1.

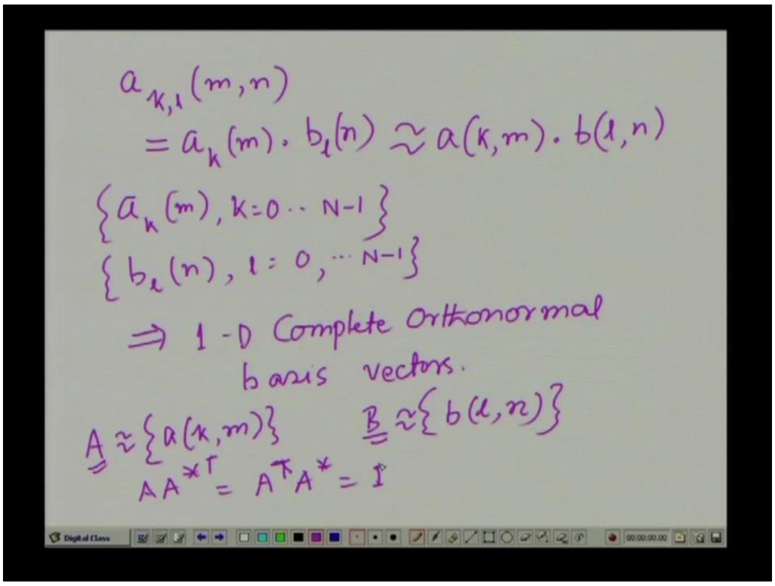
So to compute this v(k,l), you find that if I compute this particular expression. For every v(k,l) the number of complex multiplication and complex addition that has to be performed is of the order of capital N square, ok. And you remember that this has to be computed for every value of k and l, where k and l vary in the range 0 to capital N-1, that is k is having capital N number of values, l will also have capital N number of values.

So to find out v(k,l), a single coefficient v(k,l) we have to have of the order of capital N square, number of complex multiplications and additions. And because this has to be computed for every v(k,l) and we have capital N square number of coefficients because both k and l vary in the range 0 to capital N-1. So there are capital N square number of coefficients. And for computation of each of the coefficient we need capital N square number of complex addition and multiplication.

So the total amount of computation that will be needed in this particular case is of the order of capital N to the power 4, ok. Obviously this is quite expensive for any of the practical size images because in practical cases we get images of the size of say 256 by 256 pixels or 512 by 512 pixels, even it can go upto say 1k by 1k number of pixels or 2k by 2k number of pixels and so on.

So if the computational complexity is of the order of capital N to the power 4, where the image is of size n by n you find that what is the tremendous amount of computation that has to be performed for doing the image transformations using this simple relation. So what is the way out? We have to think that how we can reduce the computational complexity. Obviously to reduce the computational complexity we have to use some mathematical tools and that is where we have the concept of separable unitary transforms.

(Refer Slide Time: 14:16)



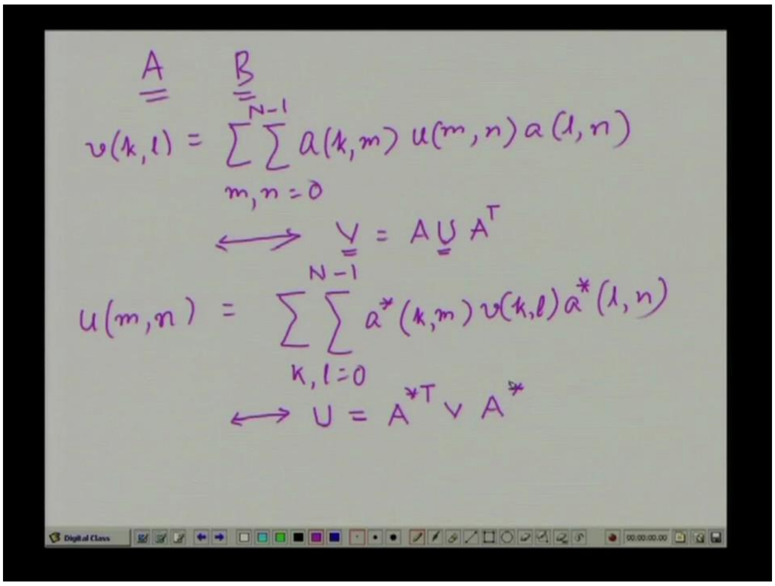
So we find that we have the transformation matrix which is represented by matrix A or we have represented this as ak,l(m,n) and we say that this is separable if ak,l(m,n) can be represented in the form so if I can represent this in the form ak(m) into say bl(n) or equivalently I can put it in the form a(k,m) into b(l,n). So if this ak,l(m,n) can be represented as a product of ak(m) and bl(n) then this is called a then this is called separable.

So in this case both ak(m), where k varies from 0 to capital N-1 and bl(n), where l also varies from 0 to capital N-1. So these two sets ak(m) and bl(n) they are nothing but one dimensional complete orthogonal sets of basis vectors. So both ak(m) and bl(n) they are one dimensional complete orthonormal basis vectors. Now, if I represent this set of orthonormal basis vectors both ak(m) and bl(n) in the form of matrices, ok.

That is we represent A as ak(m) as matrix A and similarly bl(n) the set of this orthonormal basis vectors if we represent in the form of matrix then both and both A and B themselves should be unitary matrices. And we have said that if they are unitary matrices then AA conjugate transpose is equal to A transpose A conjugate which should be equal to identity matrix. So if this holds true in that case we say that the transformation that we are going to have is a separable transformation.

And we are going to see next that how this separable transformation helps us to reduce the computational complexity. So in the original form we had the computational complexity of the order capital N to the power 4. And we will see that whether this computational complexity can be reduced from capital from the order capital N to the power 4. Now in most of the cases what we do is we assume these two matrices A and B to be same and that is how these are divided decided.

(Refer Slide Time: 18:05)



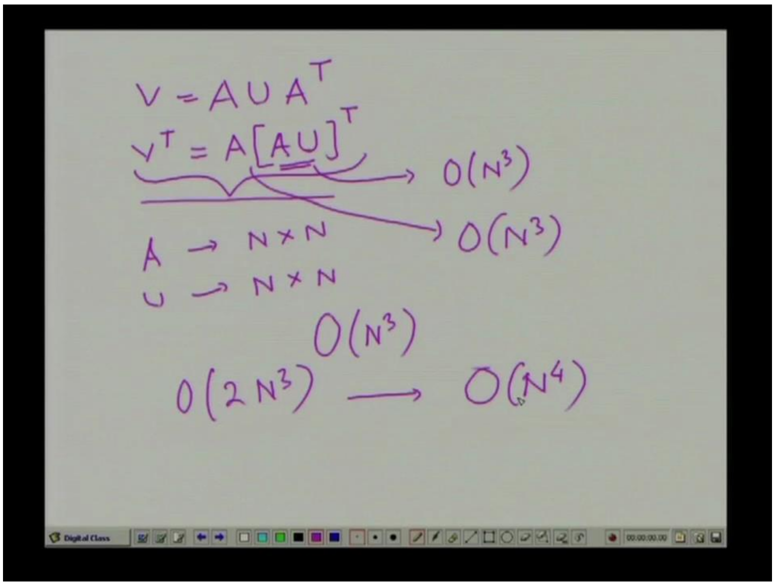
So if I take both A and B to be equal to same then the transformation equations can be written in the form v(k,l) will be double summation a(k,m) u(m,n) a(l,n), so compare this with our earlier expressions where in the expression we had ak,l(m,n). So now this ak,l(m,n) we are separating it into two components one is a(k,m) the other one is a(l,n) and this is possible because the matrix A that we are considering is a separable matrix.

So because this is a separable matrix, we can write v(k,l) in the form of a(k,m) u(m,n) into a(l,n), where again in this case both m and n will vary from 0 to capital N-1. And in matrix form this equation can be represented as v equal to AUA transpose. Where U is the input image of dimension capital N by capital N and V is the coefficient matrix again of dimension capital N by capital N. And the matrix A is also of dimension capital N by capital N.

In the same manner the inverse transformation that is what we what we have got is the coefficient matrix and by inverse transformation we want to have the original image matrix from the coefficient matrix. So in the same manner the inverse transformation can now be written as u(m,n) equal to again we have to have this double summation a*(k,m) v(k,l) a*(l,n), where both k and l will vary from 0 to capital N-1.

So this is the expression for the inverse transformation, and again as before this inverse transformation can be represented in the form of a matrix equation where the matrix equation will look like this U equal to A (tra) conjugate transpose V into A conjugate. And these are called two dimensional separable transformation. So we find that from our original expressions, we have now brought it to an expression in the form of separable transformations.

(Refer Slide Time: 21:40)



So we find that this particular expression that is V, when we have written this V equal to sorry so here we have written V equal to so if you go back to our previous slide you find that V equal to AUA transpose. So if I just write in the form AUA transpose, so I get the coefficient matrix V, from our original image matrix U by using the separable transformations.

The same equation we can also represent in the form of V transpose equal to A[AU] transpose. Now what does this equation mean, you find that here what it says that if I compute A the matrix multiplication of A and U take the transpose of this then pre multiply that result with the matrix A itself then what we are going to get is the transpose of the coefficient matrix V.

So if I analyze this equation it simply indicates that this two dimensional transformation can be performed by first transforming each column of U with matrix A and then transforming each row of the result to obtain the rows of the coefficient matrix V. so that is what is meant by this particular expression. So A into U, what it does is it transforms each column of the matrix A with of the input image A with the input image U with the matrix A.

And this intermediate result you get, you transform each row of this again with matrix A and that gives you the rows of the transformation matrix or the rows of the coefficient matrix V. And so if I take the transpose of this final result what we are going to get is the set of coefficient matrix that we want to have. Now if I analyze this particular expression you find that A is a matrix of dimension capital N by capital N.

U is also a matrix of the same dimension capital N by capital N. And then from matrix algebra, we know that if I want to multiply two matrices of dimension capital N by capital N, then the complexity or the number of additions or multiplications that we have to do is of order capital N cube.

So here to perform this first multiplication we have to have of order N cube number of multiplications and additions. The resultant matrix is also of dimension capital N by capital N. And the second matrix multiplication that we want to perform that is A with A u transpose, this will also need of order N cube number of multiplications and additions. So the total number of addition and multiplication that we have to perform when I implement this as a separable transformation is nothing but of order 2N cube.

And you compare this with our original configuration when we had seen that the number of addition and multiplication that has to be done is of order N to the power 4. So what we have obtained in this particular case is the reduction of computational complexity by a factor of capital N. So this simply indicates that if the transformation is done in the form of a separable transformation, thank you.