

# Digital Voice & Picture Communication

Prof. S. Sengupta

Department of Electronics and Communication Engineering

Indian Institute of Technology, Kharagpur

Lecture - 22

Embedded Zerotree Wavelet Encoding

Last class we covered that how to use the discrete wavelet transform in images, then we had also planned to cover that how the DWT coefficients are actually encoded in order to generate the bit stream. Now we could not exactly cover to the extent we had decided in the last class because of some shortage of time, so we are going to continue with that in this lecture. The title that we have for this lecture is embedded zerotree wavelet encoding.

Now, towards the end of the last lecture I had actually introduced to you the concept of the parent-child relationship that exists between the coefficients in the different subbands and especially we had seen that whenever we are changing from one resolution to the next; to the more final resolutions whenever we are going, there we are finding that one pixel or one coefficient in the coarser resolution or coarser scale that corresponds to four coefficients in the next final level of scale and this is what will form a kind of a tree where the root node would be the coefficient that belongs to the top most level of the LL subband.

So we are essentially assuming a dyadic partitioning where the LL subbands are successively split into four more subbands and each LL is iteratively sub divided further. Then we had seen how to obtain that kind of a data structure and essentially that data structure exploits the coefficient the relationship that exists between the coefficients in the different subbands and this is what we are going to exploit in the encoding of the DWT coefficients.

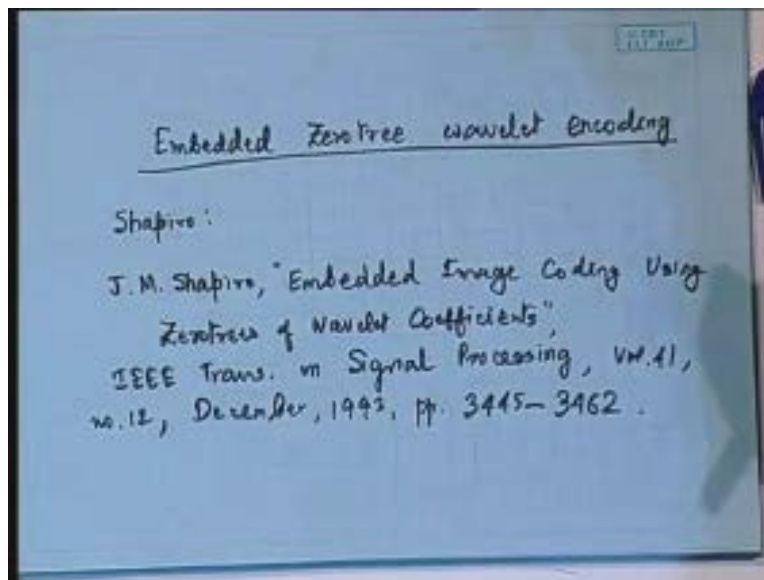
To encode the DWT coefficients actually we are first going to discuss about a very popular technique which was proposed by Shapiro and there is a very classic paper written by Shapiro. Let me give you first the reference of the paper so that you can yourself go through and find out details which are more than what we will be able to cover in the class actually. Because in the

class I will be giving the essence of the algorithm in a nutshell way but more details of it you will find in this paper.

The paper is written by Jerome M. Shapiro is the Author of the paper and the title is Embedded Image Coding using Zerotrees of Wavelet Coefficients.

Now, the term zerotree is not yet known to us; tree is known because we have just described what a tree structure is going to be **in terms of the wavelet coefficients** in terms of the zerotrees of the wavelet coefficient and as I was telling you this is a classical paper and it appeared in IEEE transactions on Signal Processing. You can find it in volume 41 number 12 December 1993 and the page numbers are 3445 up to **let me tell you the last page of the paper** it is 3462 so the paper is actually 17 or 18 page paper, fairly a long one, but after you listen to this lecture it will be quite instructive to go through this paper and have a more detailed feeling.

(Refer Slide Time: 6:13)



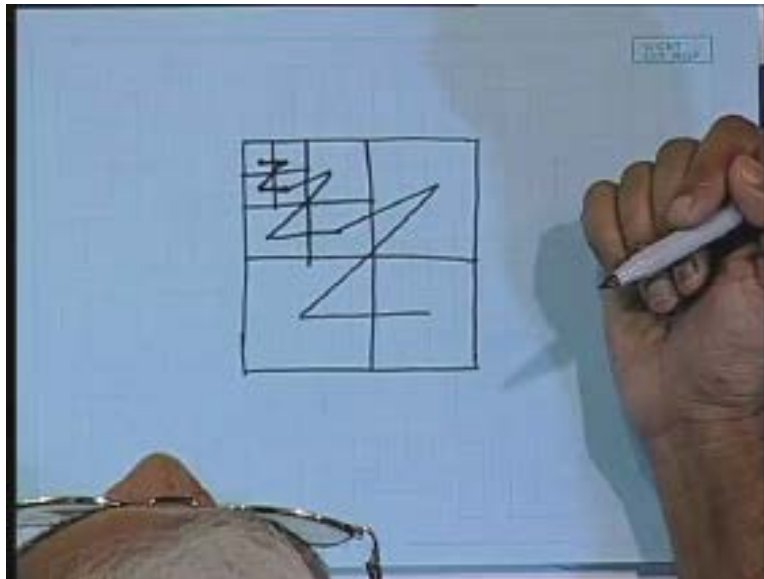
First essential philosophy behind this embedded zerotree coding is, you see that what our objective is; we want to achieve compression and to achieve the compression as you know that

we have to exploit the redundancy and in this case the redundancies are exploited between the coefficients which are there in the same pixel location but in the successive subbands.

What I mean to say is that in the last class you had seen that if we have a dyadic partitioning like this; so if this is the first level of decomposition and then we go through the second level of decomposition and third level of decomposition and so on so what we do is that we pick up the coefficients from the top most LL subband which is this one and then we form a tree where the descendents would be first the HL LH and HH coefficients in the same resolution and we will be picking up those coefficients in this order. That means to say that first start with this as the root node, go over to this explore this, pick up this the corresponding coefficient in this subband, pick up the corresponding coefficient in the next subband; corresponding means corresponding spatial position, because this coefficient which you are picking up from the LL subband is having a coordinate in the LL subband so have the same reference coordinate in all the successive subbands in this order of coverage.

So the order is LL HL LH HH, then you go over to the next HL subband. Here you have to pick up the four coefficients because each coefficient here would correspond to four coefficients here then pick up this four coefficient, pick up this four coefficient, (Refer Slide Time: 8:38) pick up this sixteen coefficient, this sixteen coefficient and this sixteen coefficient. Now at the same spatial position we have picked up the coefficients at different resolutions, at different scales of wavelet filtering.

(Refer Slide Time: 8:56)



Now why are we doing that?

As I was telling you that we will be finding that most of the coefficients are highly significant in the low frequency subbands. That means to say that in the LL we will be finding almost all the coefficients to be highly significant and as we go on to the finer and finer resolution and that too in the high pass filtered subbands we will be finding lesser and lesser significant coefficients. So there is a possibility that we will be picking up considerable number of insignificant coefficient or zeros when we traverse down this tree.

Now there is a great trick which has been suggested by Shapiro and that goes like this that, since you are traversing in a tree like this you form a tree first and then whenever you are traversing a tree in this order, if you are finding that a particular coefficient at the higher level of subbands that means to say that here (Refer Slide Time: 10:21) if you find that any coefficient is insignificant let us say zero and if you find that all its decedents in the tree are also zero in that case there is no point in encoding each and every zero coefficient; you can say that okay, beyond this everything is going to be zero, everything in this tree is going to be zero and that is what forms a zerotree; zerotree means where zerotree is a part of a tree structure where all the elements starting with some intermediate level of parent some intermediate parent node will have

its own value as zero and all the subsequent decedents to it as zeros. If you find some kind of a node like that then that node is defined as a zerotree node and the embedded zerotree wavelet coding essentially makes use of that philosophy. That is the origin of the name zerotree.

And why is it embedded?

You see that whenever we are encoding the coefficients, we have to cater for what; we have to cater for certain bit rate that the channel in which we are transmitting the bits the encoded bits that channel is having some limited bandwidth. Now one may have some a priori idea about that channel bandwidth, one may not have a priori idea of channel bandwidth and even if one has a priori of channel bandwidth it is so possible that it changes dynamically; at some point of time you may get higher bandwidth at some point of time you may get lower bandwidth; you must have experienced while accessing the internet, you must have experienced that sometimes you are really feeling very happy that you are getting excellent internet speed so you are telling your friends that yeah today the net is really excellent I am downloading everything in a flash and then within five minutes time you may find that your computer is not giving any response because by then the internet sites have become slow and you are having very less bandwidth may be that at that point of time you will find that it is downloading at 1 kilobit per second rate and sometimes it may be downloading at 100 kilobits per second or even higher rates than that.

All that I want to say is that, even whenever you are making some **image transmission over the net** encoded image transmission over the net these things can happen. So in that case what one has to do is that one has to adjust the quality of the image accordingly. Well, if the bandwidth is higher in that case give a good quality; if the bandwidth is a poor feel satisfied with a cruder quality of the image. So, in order to do that what one does is to go in for an embedded coding.

The philosophy is that you generate the bit-stream in a prioritized order. By prioritized order I mean to say that judiciously you select that which coefficients are very much significant, so more significant coefficients you transmit first and then you transmit the lesser significant coefficients. The beauty of this scheme is that, well, if your bandwidth is good you can go till the end, you can encode the priority coefficients and lesser priority coefficients everything, get the best quality and if you have a very limited bandwidth after transmitting the priority coefficients if you

find that your big budget is exhausted, you do not have bandwidth beyond that so you truncate the bit-stream there.

At any point of time you should be able to abruptly truncate the bit-stream and that should be able to generate a reasonably good quality image, it would not be best quality image because we are truncating something, we are chopping off some information but it will be reasonably good. So that kind of a truncation at any arbitrary point of encoding should be possible that is the philosophy behind the embedded coding.

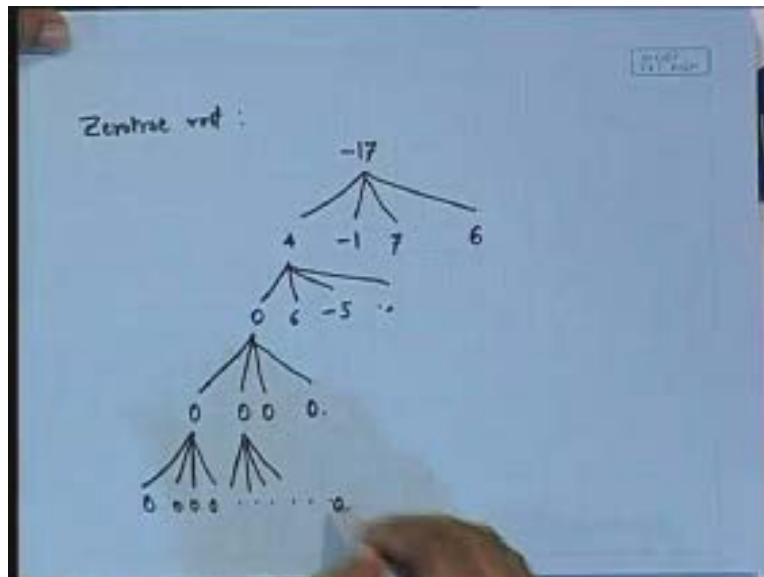
Now, because we have already gone through the discrete cosine transforms, think over could we do that in the DCTs; in the normal DCTs?

I mean, if we try to do the zigzag scanning and at some point of time, after some coefficient after some coefficients so the backward block if I find **that my big** that if my big budget is exhausted then I have to abruptly transmit truncate that and furthermore worst situation would be that whenever I have picked up the transformed coefficients and then I am doing the quantization and then encoding it using some entropic coding scheme and there, while the entropic coded bit-stream is going on, I have to truncate the bit-stream; I may lose a lot of information and my quality could be disgracefully bad. So, in order to avoid that one can make use of an embedded coding scheme like this. And wavelet, very interestingly, is amenable to such kind of embedded coding scheme as proposed by Shapiro.

Now what is the essential philosophy behind this?

Now, as I was telling you, what is after all a zerotree root; although I have said that but just a very small example; supposing I find that this coefficient value is let us say minus 17 and say its descendents are 4 minus 1 7 6; now supposing **its descendents** one of its descendants is 0 then 6 minus 5 etc.....; now here I find that when I explore this I find that all its descendants are zeros. If I explore each of these, I find that all of these are zeros; zeros means insignificant, sort of an insignificance.

(Refer Slide Time: 17:48)



Now how that insignificance is decided; that part we will come very shortly. But in this kind of a situation we would say that this particular node (Refer Slide Time: 18:06) this forms a zerotree root because beyond this there is a zerotree which is formed like this and this is a zerotree root.

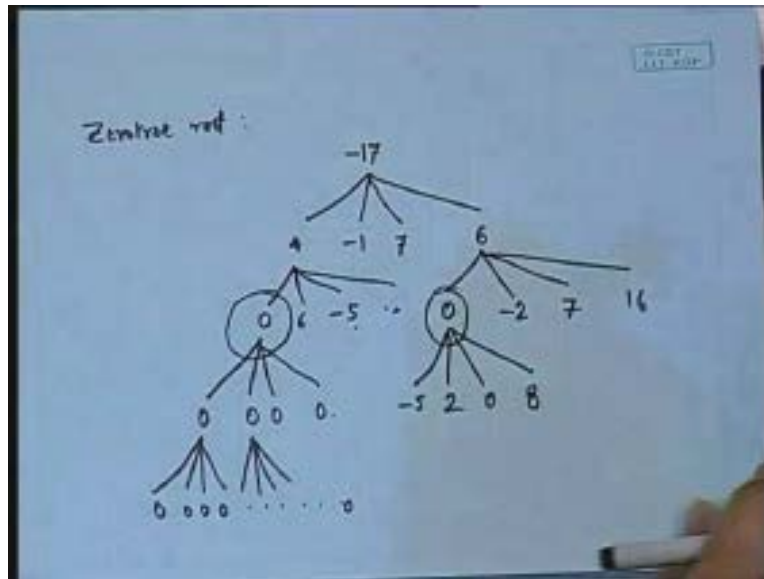
Now, zerotree root means that its values are zero and all the descendant values are zero. now let us say that anywhere here that supposing here I find that this value is minus 5 or this value is 6 let us say that here I have a value of zero and then I have minus 2, I have got plus 7, I have got 16 and supposing here its children are minus 5 2 0 8 like this now this is not a zerotree root why, because this is not forming a zerotree because beyond this, I mean, beyond its descendants we are again having some significant coefficients.

Now how is the significance decided?

The significance is essentially decided by comparing the magnitude of the coefficients. Now the coefficients could be positive or negative because you know that in the process of wavelet filtering we are using the scaling filter, we are using the wavelet filter and the values could be positive or negative and in the process of doing that we will be picking up..... I mean, because the values could be both positive and negative that is why it is a good practice to compare with

the sign with the magnitude. So if I have a coefficient value let us say capital X what we do is that we compare this value of X with a quantity T where T is the threshold.

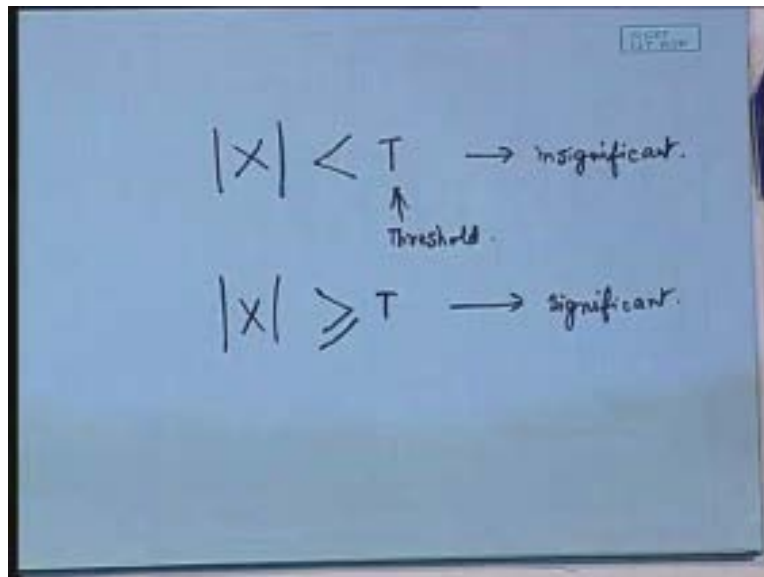
(Refer Slide Time: 19:04)



Now we define that if mod of X is less than T in that case it is declared as insignificant and if mod of X is greater than T then it will be called as significant or just to take care of the equal to case we can say that mod of X greater than or equal to T next X as a significant coefficient. Significance or insignificant could be a binary decision. In fact, I mean, that is how I should have framed the example; I had shown numerical values but you can say that rather than considering this minus 6 5 and all these things 6 minus 5 and all these things, you could take it to be all of the others to be significant and insignificant so it is a binary decision insignificant or significant but that is with respect to a threshold.



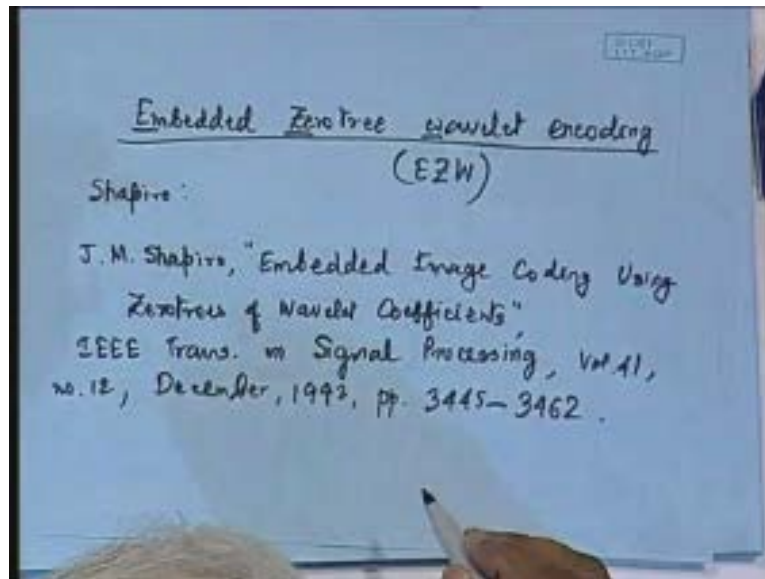
(Refer Slide Time: 21:34)



The image shows a blue background with handwritten mathematical expressions. At the top right, there is a small box containing the text "EZW". Below this, the first expression is  $|x| < T \rightarrow \text{insignificant.}$ . An upward-pointing arrow is drawn from the letter 'T' in this expression to the word "Threshold" written below it. The second expression is  $|x| \geq T \rightarrow \text{significant.}$

Very interestingly what happens is that this threshold is readjusted at every pass and readjusted in what manner? To begin with we start with a higher value of threshold and then it is a multi-pass algorithm essentially, I mean, Shapiro's embedded zerotree wavelet or what is more commonly I would rather than calling it as embedded zerotree wavelets such a long name, I would call it as EZW.

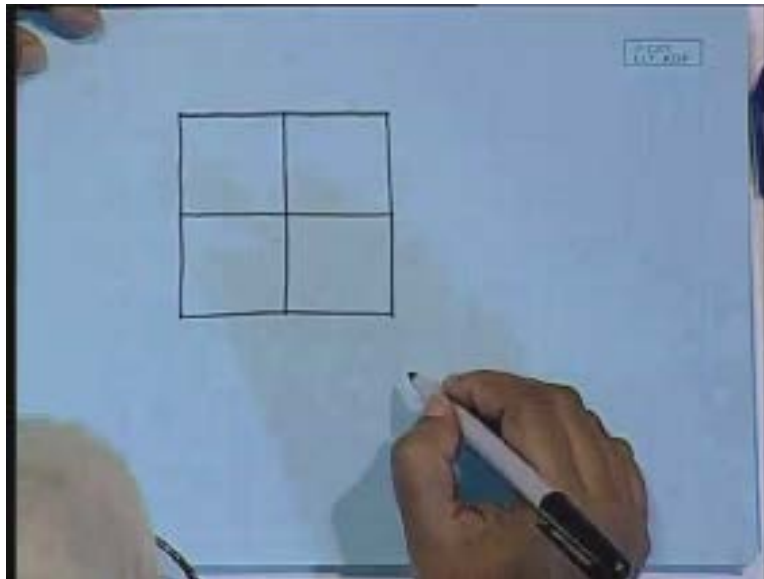
(Refer Slide Time: 21:43)



Hence it is a very popular term that you will be finding in the wavelet encoding literature about the embedded zerotree wavelet or the EZW coding technique. EZW is a multi-pass technique. So we first have a higher value of threshold so that we can decide about the significance and the insignificance in a harsh way; anything which is really having high values could be considered to be significant and anything which is less than that would be considered to be insignificant. But then what happens is that, with successive process the value of threshold is halved.

So if, say for example, that we have an array; say let us consider some example array that would be good. Let us say that we have three levels of the composition so I just pick up eight 1 2 3 4 5 6 7 8 and here 3 4 5 6 7 8 so 4 5 6 7 8 so 8 by 8 is an array that I consider; now I partition it.

(Refer Slide Time: 23:15)



So this is the level one of partitioning so that every partition is 4 by 4 and then I partition this further so that every band is 2 by 2 and this 2 by 2 is also partitioned so that we have got 1 by 1; up to 1 by 1 we can do.

Now let me write down some typical example of numerical values. So this is an encoding example and we have a value of 127 here, here we have got a value of 69, here minus 37, here minus 18 24 73 minus 18 8 44 minus 87 65 18 minus 15 21 29 minus 56 13 5 minus 8 5 minus 6 7 15 4 8 minus 11 14 minus 3 0 minus 2 3 7 (Refer Slide Time: 25:00) then come to this subband 34 38 minus 18 17 minus 27 minus 41 11 minus 5 6 17 5 minus 19 32 26 minus 7 5 come to the last band that is the HH 3 minus 9 minus 2 1 0 minus 1 0 minus 3 2 0 minus 3 1 minus 1 minus 5 7 and 4. So this is an 8 by 8 array I have just put up an example case of 64 different values.

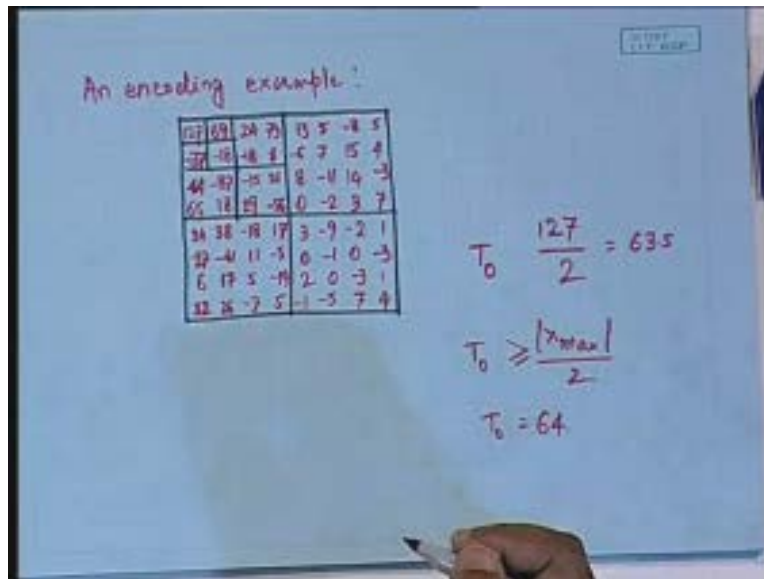
(Refer Slide Time: 25:57)

An encoding example:

127	69	24	7	13	5	-8	5
-37	-12	18	3	-4	7	15	4
44	-37	-11	11	8	-1	14	-3
65	18	19	-25	0	-2	3	7
34	39	-18	17	3	-9	-2	1
23	-4	11	-5	0	-1	0	-3
6	17	5	-19	2	0	-3	1
82	14	-7	5	-1	-3	7	4

Now if you examine this array what do you find; you find that the maximum value of the coefficients given in this example is 127; 127 is the maximum value now. What is being done is that, to start with, we normally choose a threshold which is less than half of the maximum value. So, if the maximum value in this case is 127 so the initial threshold let us say  $T_0$  should be 127 by 2 which is 63.5 so let us say that we choose a threshold equal to..... we actually make the threshold to be slightly greater than this maximum value  $x_{max}$  mod of  $X_{max}$  we have to say, mod of  $X_{max}$  by 2 so that we choose..... in this case  $T_0$  is chosen to be 64, 64 is a good number, it is 2 to the power 6 a binary number and it will be very easy to make it halved in successive passes. So we begin with  $T_0$  as 64 and then we examine this array.

(Refer Slide Time: 27:25)



Thus, what happens is that this coefficient value 127 this is a significant value. Now significant value, not only it is significant, it has got a sign the sign is positive so it is not only significant it is significant with a positive sign.

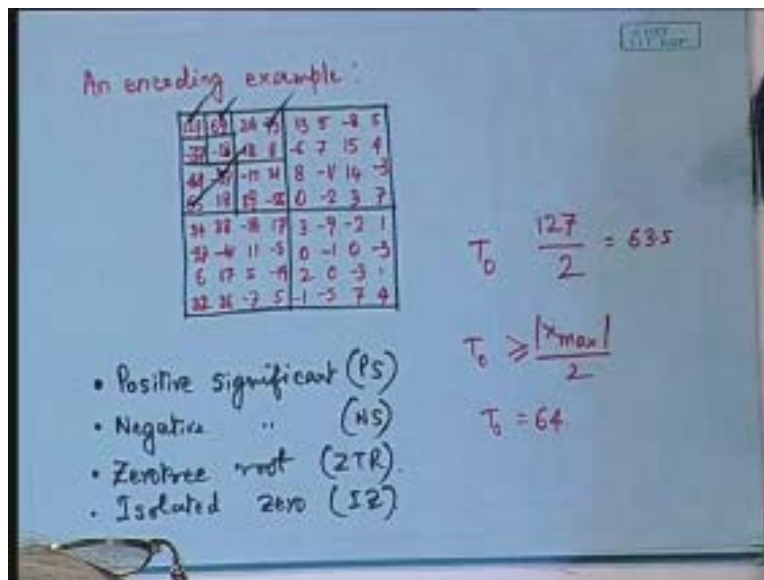
Look at the next value that is 69. This 69 value in the HL 3 subband this is also significant; come to minus 37 this is not significant because in this case the threshold is 64 and the mod of X is 37 which is less than the threshold so this not significant this is not significant again. In the HL 2 we will be finding that the 73 is significant, here in this subband what you find; minus 87 is significant, 65 is significant; any other significant value you can explore? No.

So now let us say that we come to this coefficient minus 37, this is not significant. Now is it forming a zerotree root? Minus 37s descendants are what? Minus 37s descendants are 44 87 65 18, out of this, minus 87 is significant and 65 is significant but this is insignificant. So we start with an insignificant that is 0 but its descendants are significant so it is not forming as zerotree. But look at this minus 18, minus 18 is insignificant, its descendants are these four coefficients (Refer Slide Time: 29:28).

Now again these four coefficients individual descendants are lying here, all of them are insignificant. So minus 18 is insignificant, you can call it as a zero and all its descendants are zeros so minus 18 essentially forms a zerotree root.

Now what we can call this minus 37 as; it is a 0 but it is not a zerotree root and the coefficients which are significant that could be either a positive significant or a negative significant and the coefficients which are insignificant they could be either an isolated zero means not having the descendants as zeros or they could be zerotree roots. So, for every coefficient we have got four possibilities of encoding. it could be positive significant positive significant let us say PS, then coefficients could have negative significant so this could be a negative significant NS, this could be zerotree root ZTR let us call and it could be isolated zero IZ.

(Refer Slide Time: 31:20)



Therefore, whenever we are encoding the coefficients every coefficient we have to remember that we have to encode the coefficient in one of these four categories. But the beauty is that, if we find a zerotree root, if a coefficient is having zerotree root then its descendants need not to be searched because zerotree root automatically means that beyond this everything is zero.

Now a thinking would come to you that how effective would this technique be because if we do not find such abundance of zeros then is it going to be a very efficient methodology? Very luckily yes, you will be finding lot of such insignificant coefficients and because of this you will be encountering zerotree roots. You will be finding that mostly the higher frequency subbands are having mod insignificant values and that is why it should be possible for you to detect more number of insignificant coefficients leading to more number of zerotree roots and more number of zerotree roots means that it is definitely an efficient encoding scheme because **you are** like you are only designating this minus 18 as a zerotree root, **you are not telling that** I mean, you do not have to explicitly tell any further that these values are individually all zeros or these values are individually all zeros that you do not have to do. **that is the** So there lies some kind of exploitation of the redundancy and this is what Shapiro's technique has made.

In fact I did not complete this example. **but you can definitely go through yourself**; take an example array like this and then you can find out **that what are the** that how every coefficient is going to be treated like this.

(Refer Slide Time: 33:59)

An encoding example:

10	15	24	15	13	5	-8	5
22	-9	12	8	-4	7	15	9
44	10	-15	21	8	-8	14	-3
12	18	19	-25	0	-2	3	7
34	28	-12	17	3	-9	-2	1
22	-4	11	-3	0	-1	0	-3
6	17	5	-19	2	0	-3	1
32	25	-7	5	-1	-3	7	4

- Positive significant (PS)
- Negative " (NS)
- Zerotree root (ZTR)
- Isolated zero (IZ)

$$T_0 = \frac{127}{2} = 63.5$$

$$T_0 \geq \frac{\max}{2}$$

$$T_0 = 64$$

$$T_1 = 32$$

Now, when the first pass is completed then in the next pass we are going to make the threshold let us call it as  $T_1$  which should be 64 by 2 or rather 32. Next step it will be further half so  $T_2$  would be 16, next step it would be  $T_3$  is equal to 8 and so on. Now why do we make that?

To start with, we have put a very high threshold. So anything which is less than 64 is insignificant. But with higher and higher passes we are bringing down our significant definitions. So we are very strict at the beginning but we are becoming more and more lenient as time progresses.

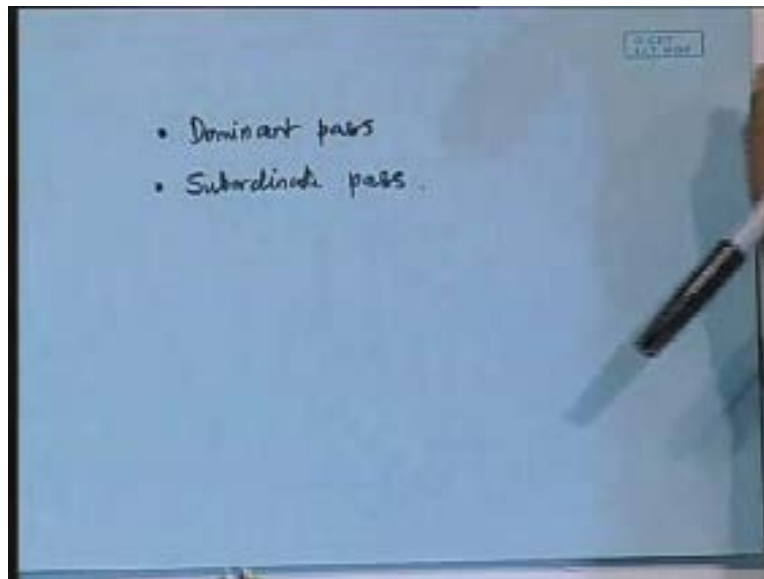
What is the essential philosophy of that?

Because if you have to generate an embedded bitstream better start with a strictness in the sense that only whatever are very significant, I mean, at the end of the first pass you will be finding that only a few coefficients you would pick up as significant and those will be encoded rest several coefficients would be treated as zerotree roots and we do not have to encode that. But if we are having more number of passes that means to say that if our bid budget permits several pass encoding in that case we will come to the next level of encoding and then we will be able to encode more and more coefficients. If we still have bits left we can go in for the level  $T_2$ , we can come in for level  $T_3$  and so on.

Now, in this kind of multi-pass encoding every pass consists of two things: one is what is called as the dominant pass; each pass is subdivided into a dominant pass and a subordinate pass.



(Refer Slide Time: 36:07)

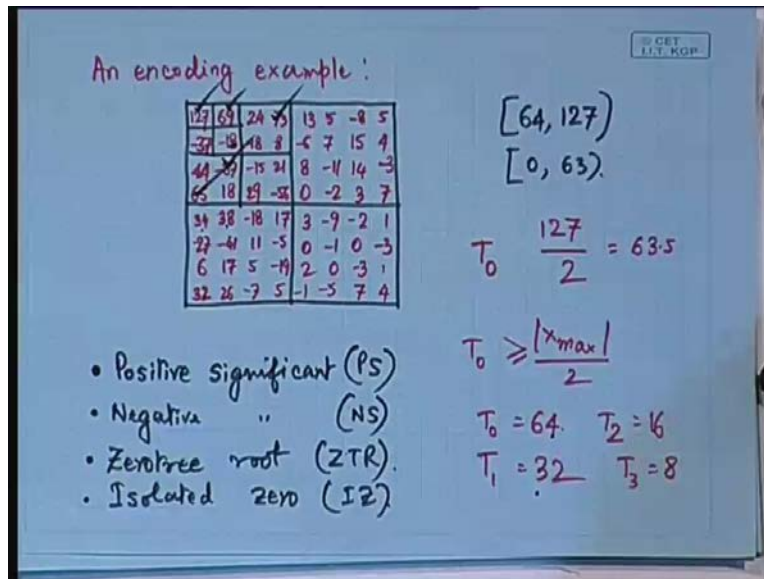


By dominant pass we mean the coefficients which are yet to be found to be significant. So **coefficients so** dominant pass encodes the coefficients which are yet to be significant. What I mean by this is, let us take this example again (Refer Slide Time: 36:38); you see this coefficient minus 37 is insignificant when the threshold is 64 but in the next pass when the threshold is 32 then this minus 37 become significant. So whichever are lying between the value of 32 and 64, like here this becomes significant, this becomes significant, this 34 become significant there are quite a few values which become significant in the second pass when I make  $T_1$  is equal to 32.

So dominant pass picks up or encodes the coefficients which are..... or rather coefficients which are yet to be significant that are identified and they are examined to be isolated zeros or zerotree roots whereas the coefficients which are just found to be significant they have to be encoded as positive significant or negative significant. And then there is a subordinate pass and what that subordinate pass does is that coefficients which are already found to be significant in the earlier pass, their values or their magnitudes are refined. So in the subordinate pass, coefficients that are already found to be **already found to be** significant are refined in magnitude and how do we do that, it is by a process of successive approximation.

What I mean by successive approximation is like this.

(Refer Slide Time: 38:43)

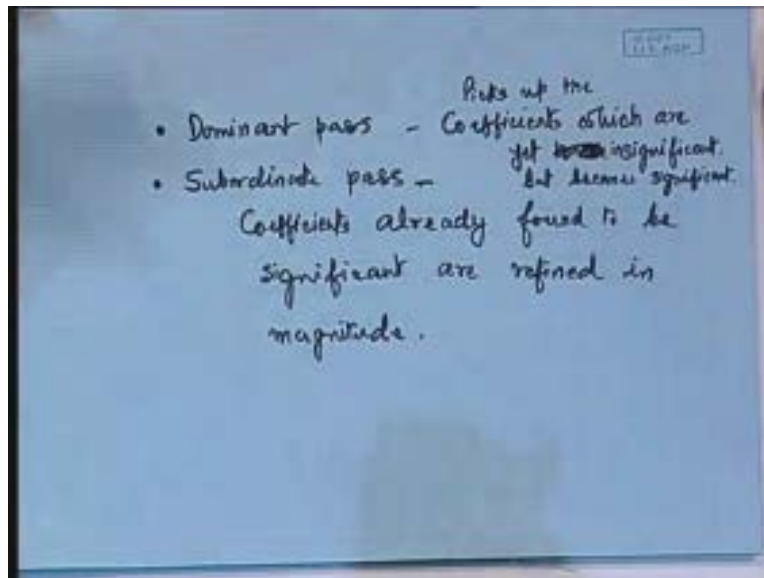


Let us say that we have value equal to 127. Now initially in the first pass when this value is 64 the threshold value is 64 it is significant. So we say that this is encoded as a positive zero positive significant so it is having a value equal to 1. Now 1 means what; we know that it is in the range of 64 to 127 because 64 is a threshold and we know that the maximum magnitude here is 127 so it is lying somewhere in the range of 64 to 127, this 69 also lies in the range of 64 to 127 and whatever is significant that lies in the range of 64 to 127 and whichever are insignificant they lie in the range of 0 to 63. So in this case what happens is that we are just subdividing into two such parts. Now whichever is within 64 to 127 we can encode them as some value in the middle of the range let us say we can encode it as 96 and 0 to 63 we can encode it as 32.

Now, next pass we identify that this 64 to 127 range we just find out that whether it will be in the range of 64 to 96 or whether it is in the range of 96 to 127 or 64 to 95 or it is 96 to 127 and then we approximate the value by a value equal to middle of that range. So like this there is a successive approximation which is done in its magnitude encoding and that successive approximation is carried out in the subordinate pass and the dominant pass basically picks up

picks up the coefficients which are yet insignificant, I mean I should modify the language; dominant pass picks up the coefficients which are yet insignificant but becomes significant in this pass but becomes significant in the current pass.

(Refer Slide Time: 41:23)



This is the way whereby the dominant pass will be followed by subordinate pass and then again we will be going we will be reducing the threshold by a factor of 2 and we will be once again carrying out a dominant pass and subordinate pass, again we will keep on repeating this.

Now the basic idea about the successive approximation technique of encoding which is being done in this subordinate pass is that, well, if you have a big budget you can more accurately represent the coefficient and if your big budget is poor in that case your value representation is true, you have to accept that.

So, if you can tolerate only a single pass then a coefficient anywhere lying between 64 and 127 will be encoded as 96 the middle of the value and if there is any coefficient which is and if we can go through the second pass then at least we can more accurately represent whether 64 to 95 or 96 to 127; if you go in for another pass then we can represent it even better. So this is

something like you have seen that in the bit plane encoding also we have got a very similar philosophy. That means to say that start with the most significant bit plane and then progressively go in for the lower and lower significant bit planes and coming down to the level of the least significant bit plane. So this way of embedded zerotree wavelet it is very efficient technique and because there is abundance of such zerotrees; especially at the beginning of the passes we will be having more presence of zerotree roots that is why their encoding can be done very efficiently.

Although it is a very attractive technique and in fact one should be indebted to Shapiro for proposing such a very nice scheme which all subsequent researchers had accepted this and this became a very standard technique for wavelet encoding, but there are certain limitations if you try to observe it very closely. One of the limitations is that such a kind of encoding scheme preassumes that you are only going in for a diadic partitioning meaning that you are only splitting the LL subband iteratively.

Now, if you happen to split up any subband other than the LL to have some analysis then this technique does not work. And another limitation is that it is exploiting the redundancy which is present at a particular position at a particular spatial position but across different scales that redundancy is exploited by having a very efficient scheme in employing this zerotree root concept. But it is not considering or it is not exploiting the redundancy which is present within the coefficients within the subband, I mean, the coefficients which are forming some spatial neighborhood so their neighborhood similarity of coefficients that is also not explicitly exploited.

Therefore, subsequent to Shapiro's paper as I told you in the reference itself that Shapiro's paper was published in the year 1995 and it was subsequently refined further by Said and Pearlman so they had come up with some better scheme which is called as the set partitioning in hierarchical trees set partitioning in hierarchical trees. This is called as SPIHT. And after the that So that was published in I think 96 or 97 unfortunately I do not have the ready reference with me right now but subsequently SPIHT there was an effort by the research community to develop a standard an encoding standard where the discrete wavelet transform can be used.

Now, we have been so long discussing about the still image compression standard and in still compression standard, as I mentioned, the first standard that was proposed is the JPEG standard and still even today JPEG standard is continuing to be used as a popular technique. But looking at the futuristic requirements and especially the requirements for encoding very large size images which will come with a very high resolution or high definition television signals HDTV applications or where the picture resolution will be much higher, there, the JPEG techniques would have some limitations; limitations in the sense that one needs a better compression efficiency as compare to the JPEG. So with that in mind **the standard for** the next version of the standard was designed and that is JPEG 2000.

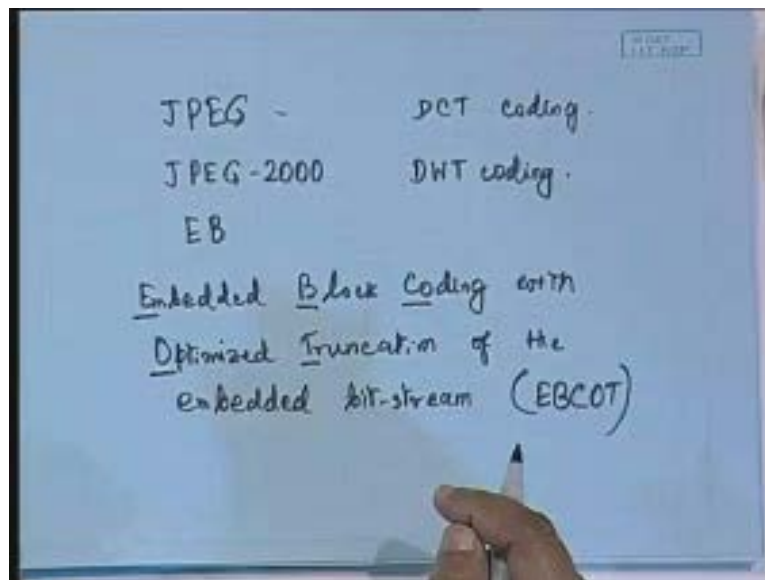
As the name implies that JPEG 2000 was addressed..... the work for JPEG 2000 began with an objective that it should be completed by the year 2000 to formulate this next generation of standard for still image compression. There was some delay so I think that by the end of 2000 or beginning of 2001 the standard was proposed. JPEG 2000 has used the discrete wavelet transform coding, so this has used DWT coding whereas as you already know that JPEG had used DCT encoding.

In JPEG 2000 the DWT encoding how was it used?

One approach could be to use the PZW; I mean the technique which we already described. But because of its limitation a better technique was explored. But yet the people who framed the standard, they had kept one aspect in mind, getting the inspiration from Shapiro's algorithm that even the new standard which is going to be designed there one should have a very or rather one should adopt an embedded encoding scheme. So the embedded coding scheme was not dropped, Embedded approach was still considered but the encoding instead of doing it in the Shapiro's zerotree wavelet way it was done by considering the significance within the subbands; it was also a multi-pass kind of a subband but it was done with an objective **that different** that rather than dyadic partitioning any other way of partitioning also can be employed and then the **redundancy is present within the coefficients** redundancies present for the coefficients within the subband that was also exploited and the technique that was adopted there is **embedded bit-streams or embedded block coding** embedded block coding **let me write down the words fully** embedded block coding with optimized truncation of the embedded bit-stream, a very long word,

a very long sentence it is but every time we talk in terms of abbreviations, embedded block coding, the basic philosophy is optimized truncation and this is called as EBCOT a short form or EBCOT.

(Refer Slide Time: 52:25)

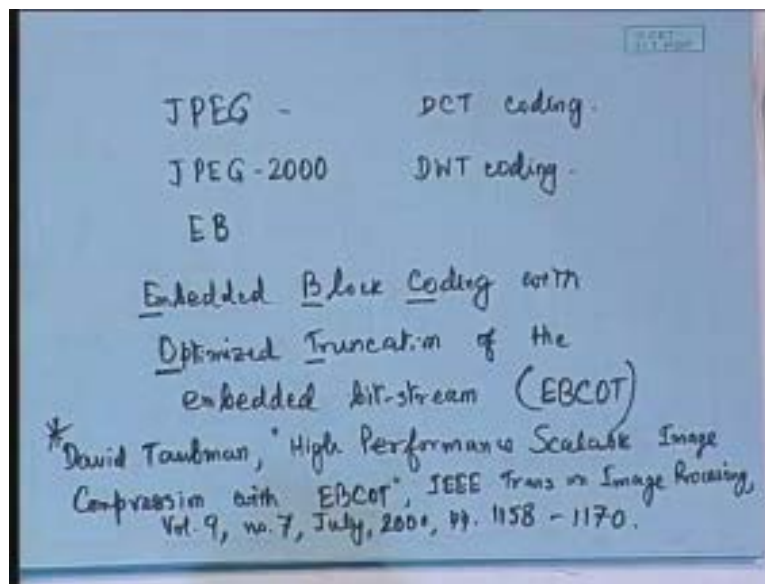


Now embedded coding we are familiar with because Shapiro's technique is also a kind of embedded coding. Now here it is called embedded block coding because the subband, the samples within the subband or the coefficients within the subband they are subdivided into blocks and it is encoded block-wise so there is a block coding and the beauty behind this technique is the optimized truncation. Why optimized truncation is because even in Shapiro's algorithm it is said that you can truncate the bit-stream and when you truncate the embedded bit-stream you are going to lose information but till the point where you have truncated at least its quality should be preserved. But the truncation point could be anywhere and at least near about the truncation point you are going to lose some information which is going to create some error in the reconstructed image.

Whereas in the case of EBCOT they adopt an optimization technique, a red distortion optimization technique so as to specify some specific points of truncation, some specific points

at which the bit-stream may be truncated. So it is a kind of a quality versus bit rate tradeoff which is specified because it is going into a standard and the man behind this EBCOT or embedded block coding with optimized truncation of the bit-stream is David Taubman and there is yet another classic paper by David Taubman which you should read. The reference of that is, Author is David Taubman **Taubman Taubman** "High Performance..... the title of paper is "High Performance Scalable Image Compression with EBCOT" the paper appeared in IEEE transactions on image processing, volume 9 number 7 July 2000 page 1158 to 1170.

(Refer Slide Time: 55:50)



This is also a very important paper, you should read that. This EBCOT algorithm is actually the one that has gone into JPEG 2000; EBCOT in little bit of modified form. I mean, what Taubman proposed is slightly different from what finally the JPEG 2000 had adopted.

Now you may find the initial reading of the paper little difficult because of the lot of details and intricacies which are there in the EBCOT but if you patiently read that paper you will be developing the interest and you will come to know that how the **JPEG 2000 is** JPEG 2000 bit-stream is encoded. The essential advantage behind this EBCOT algorithm is that this permits one

to very efficiently encode the region of interest and as I was telling you already the scalability tradeoff, the quality bit rate scalability can be very nicely achieved using this EBCOT algorithm.

**I am not going into the details of EBCOT algorithm**, just to make a mention and the rest is up to you to study it further, just to give you some direction that where we have in today's technology. So, in the next lecture we will be going over to the video compression because so far we have covered all that I wanted to talk on the still image compression; in still image we have covered the DCT and DWT as techniques and as standards; there are only two standards the JPEG and JPEG 2000, but we will describe the basic philosophy behind video coding in the coming lecture, thank you.