Digital Voice and Picture Communication

Prof. S. Sengupta

Department of Electronics and Communication Engineering

Indian Institute of Technology, Kharagpur

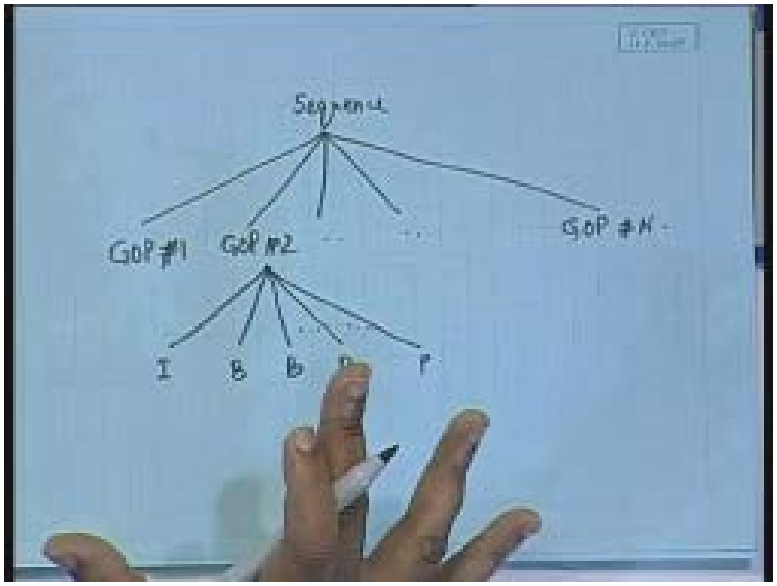
Lecture - 26

Video Coding Standards: H dot 264

.....the video coding standards and instead of spending time with the earlier standards we will just make a very quick reference and go over to the H dot 264 which is the latest in the standards. So I will be covering mostly H dot 264 in details in this lecture and just before coming to H dot 264 we have to make some passing reference about some of the concepts of video coding which has been already done in the previous standards. So we will take some reference of the previous standards and then talk about this.

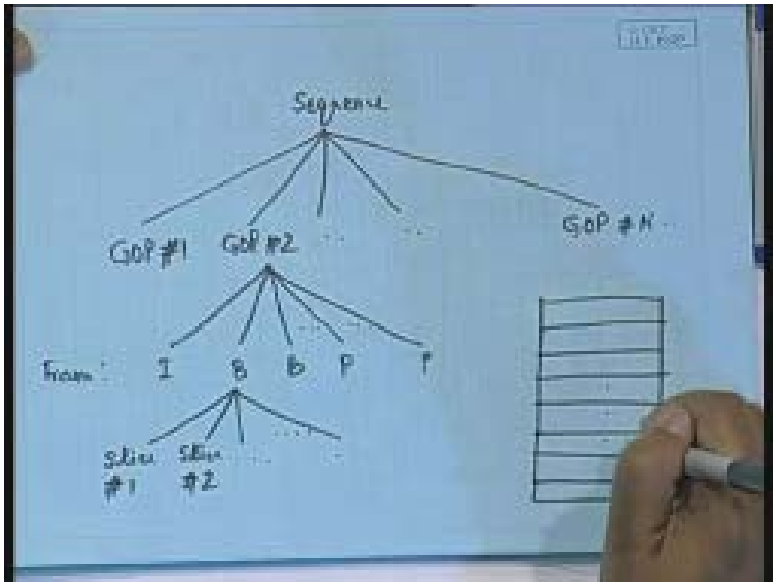
Let us now come to the one or two very important aspects of the video coding. You see, we have already told you, that basically when you have a video sequence, then the sequence is essentially partitioned into a group of pictures. We can call those groups of pictures by the name GOP. So we start with the GOP number 1 then GOP number 2 and so on so there will be a group of pictures which I had said that will consist of..... I mean, if you just take the composition of any GOP you will be finding like this that there will be an I frame followed by there will be some B frames there will be some P frames in between again it will have some P frames and more number of P frames and B frames in between.

(Refer Slide Time: 2:54)



This is how the GOP structure will be there. This also we have discussed. Now, every such frame these are all the frames individual frames; every frame is actually partitioned into a set of slices. Let us say we just look at any particular frame where we will be having this frame partition into various slices like slice number 1, slice number 2 like this where slice will be a part of the picture.

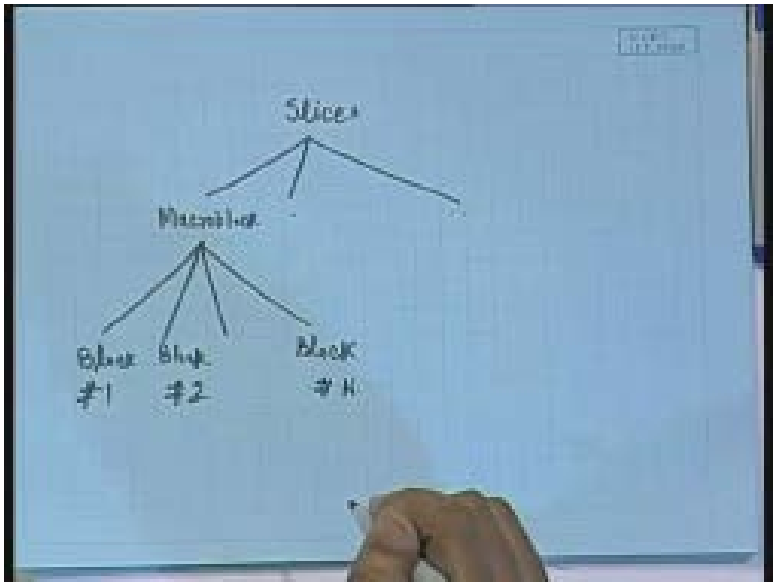
(Refer Slide Time: 3:47)



Like say for example, if this is the frame then one can divided into let us say 3, 4 say this is the picture so 3 4 5 6 7 8 so eight slices we have partitioned. Like this, slices are actually some subparts of the frames for which a specified data structure is defined.

When we look at the composition of the slices the slices are composed of some macro blocks. And what is macro blocks; macro blocks are basically composed of a group of blocks. So macro block actually is partitioned into blocks and block size you already know that block size; for all the initial standards we have been taking the block size to be 8 by 8 but only in the most recent one we are having the block size as the variable and it can be as low as 4 by 4 block sizes.

(Refer Slide Time: 4:43)

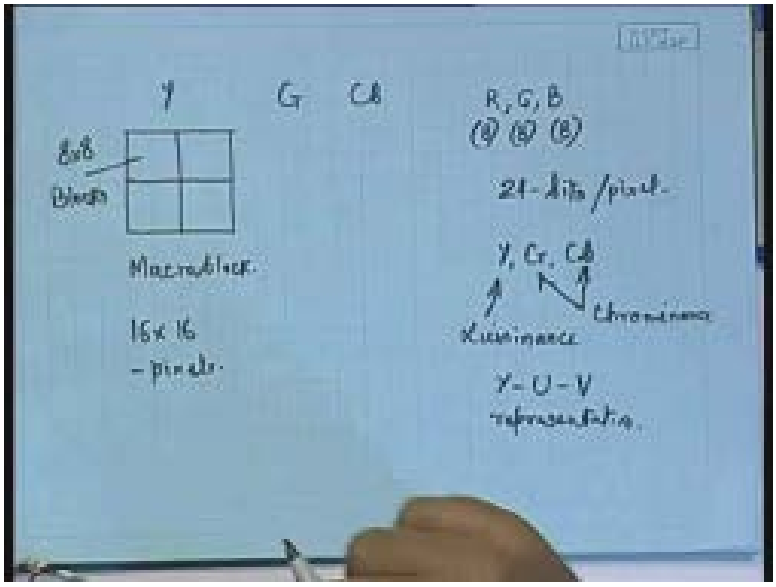


Macro blocks are composed of blocks and I will now tell you that how these blocks and macro blocks are related. Basically if I take every block to be let us say 8 by 8 size then what one can do is that whenever one is having a colored image we take..... see, the color image can be represented in two formats: one format is the RGB. In the case of RGB format what we do is that we allocate 8 bits to each of these color planes: 8 bits for R, 8 bits for G, 8 bits for B where we will be having essentially 24 bits per pixel. But for the RGB the information pertaining to the color as well as of the intensity are all embedded within, there is no separate information for the color or no separate information for the intensity. The R, G and B values themselves contain the intensity as well as the individual color planes.

Now there is an alternative representation which is done and that representation is the Y, C r, C B structure where Y is the luminance which basically indicates the intensity values and C r and C B they indicate the chrominance values. So the luminance chrominance representation's advantage is that, luminance chrominance information actually takes you to the intensity and the color parts as separate ones.

Now what happens is whenever we are having such luminance chrominance representation Y, C r, C B representation or it is sometimes referred to as the YUV representation, in the case of Y-U-V representation we take a 16 by 16 block size a group of 16 by 16 pixels where let us say that it is like this (Refer Slide Time: 7:26) and this 16 by 16 size which we call as macro block every macro block will consist of a luminance part of 16 by 16 pixels. It consists a luminance of 16 by 16 pixels and if our block size is of 8 by 8 then every luminance macro block is going to have four blocks so I can partition a luminance macro block into four luminance blocks so each of these will be of size 8 by 8 these are the blocks and we have four such blocks. And other than the luminance we also have the color components, the C r and the C B.

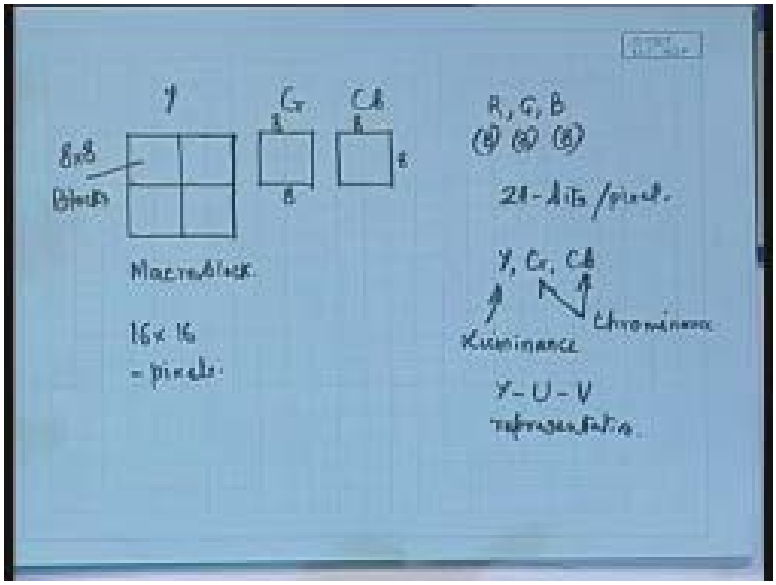
(Refer Slide Time: 8:16)



Now it is seen from the psychovisual experiment that our eyes are more sensitive to intensity as compared to color and as a result of which we need to preserve the resolution in the intensity or the luminance space and we need not have to keep that high resolution for the color spacing comparison and the color components can in fact be subsampled by a factor of 2 in both horizontal as well as vertical directions meaning that only one fourth of the size as compare to the luminance values are enough for the viewing of the color components of the picture so that corresponding to a 16 by 16 area in the luminance image we can subsample the pixels by a factor

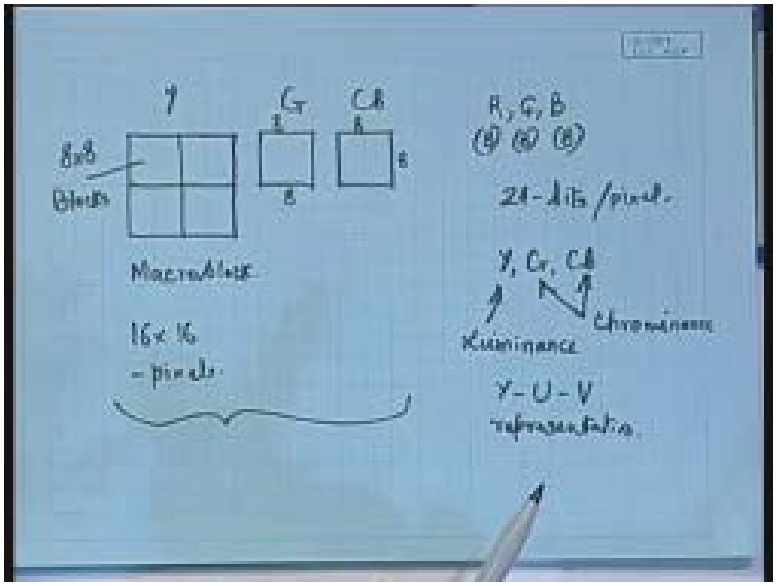
of 2 in horizontal and vertical direction and by subsampling this 16 by 16 will be converted to an 8 by 8 chrominance and it will also have an 8 by 8 the chrominance red and chrominance blue components. So we have essentially six blocks; six blocks of size 8 by 8, four blocks corresponding to the luminance and one block corresponding to C r one block corresponding to C B and this we can only do because the because in the color space we can subsample by a factor of 2 without effecting the psychovisual performance.

(Refer Slide Time: 10:14)



Therefore, making use of this psychovisual redundancy aspect it is possible to have a representation like this. This is the relationship between the blocks and the macro blocks. This whole thing will be a macro block consisting of six macro blocks of the Y, C r, C B image consisting of six different blocks; four of luminance and two of chrominance like this.

(Refer Slide Time: 10:37)



Therefore, having known this structure now it is very important for us to see that how do we encode every macro block. First of all that why are we having this type of a structure. You see, whenever we are encoding a video frame, there what are we doing; essentially the important thing what we do is to quantize the transformed samples. So whenever we are quantizing then it is very important to note that what should be the extent of quantization or rather there should be a quantization parameter which should be specified. Means, if we have a quantization matrix in that case what quantization parameter we use that will depend upon the coarseness of the quantization; whether we want coarse quantization in which case we have to use larger values of the matrix, we have to multiply the matrix elements quantization matrix elements by some constant factors whereas if we want finer quantization then we need not have to multiply this. Hence, this quantization parameter, this can vary from macro block to macro block because it is a 16 by 16 area.

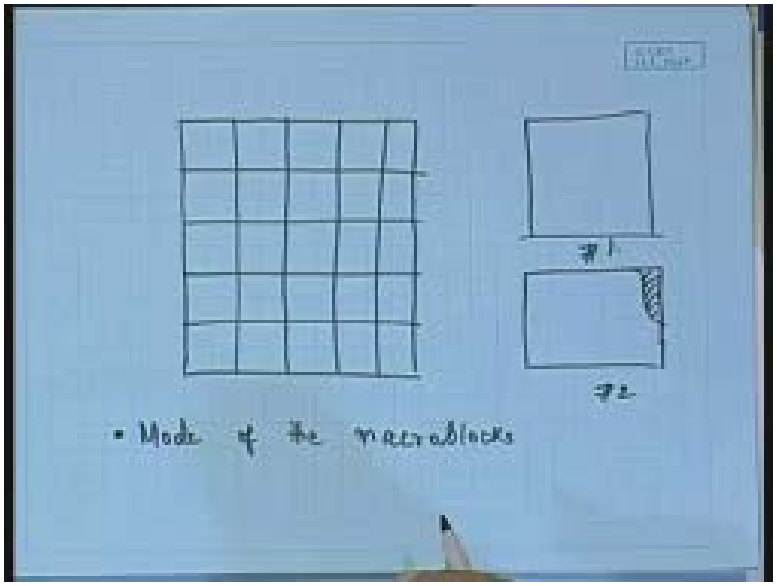
Therefore, if we have an image let us say that this is an image what we have so what we do is that this image is partitioned into a group of such macro block. These are all the macro blocks to which the frame is subdivided and within this macro block we will be having the blocks in the Y, C r and C B planes.

For now, whenever an image is there it is not that the details are present uniformly over the image, there are some parts of the image where more details are present where we have to encode that with a better quantization factor finer quantization factor there are smooth regions of the image where we need to encode with coarse quantization factor will be good enough. Even whenever we are encoding the motion also or rather the motion predicted image whenever we are encoding the difference between the incoming and the motion predicted image there also wherever more amount of motion is involved there we have to do finer quantization and wherever less motion is involved we have to do coarser quantization.

That is why in essence what I mean to say is that, from one region of the image to the other the extent of quantization can vary and that is why each macro block is going to have a different quantization parameter. Not only that, not only the quantization parameter for every macro block is important, one also has to specify the mode of the macro block. By mode of the macro block I mean to say that; as I was mentioning that there are some macro blocks which do not undergo any significant change but there are some macro blocks which are undergoing very significant change.

For example that supposing we have an image where in frame number 1 we do not have a particular object but in frame number 2 a particular object has appeared; say this is the part of an object which has appeared in frame number 2 but it was not there in frame number 1 (Refer Slide Time: 14:36).

(Refer Slide Time: 14:39)



Now that means to say what?

That means to say that in frame number 2 we have got some new information which has come in. So what if we try to do it as the simple interframe coding. Taking the frame 1 as reference how about doing the motion estimation and finding out?

Well, for those areas where it was visible where the object was visible in frame number 1 and in frame number 2 it is displaced to some extent yes, for that you do not have for those parts of the image you do not have any problem in finding out the motion vector. But for this region where you are having some new information how can you obtain correspondence? You will try to obtain some correspondence, you will try to obtain some minimum form of absolute difference and find out a motion vector but that motion vector is going to be erroneous definitely which means to say that if you are trying to predict an image based on that motion vector you will be observing large amount of error large amount of prediction error.

Now depending upon the prediction error from one macro block to the other you can decide that whether that particular macro block you are going to encode at all with interframe coding or whether you are going to treat that macro block treat that part of the image like intra; intra means without any past reference frame.

Therefore, in the mode on the macro block you can define some macro blocks as coded in the intra mode and there will be some macro blocks for which you are finding the correspondence in the previous frame you can encode them in the interframe. So there is intraframe prediction, there is also interframe prediction for every macro block. Hence, it is not essential that whenever we are encoding a particular frame as the P frame or B frame then all the macro blocks pertaining to that picture has to be inter-coded; never think like that. the frame can be a B frame, the frame can be a P frame but some of the macro blocks could be intra-coded macro blocks some macro blocks majority of the blocks may remain as the inter-coded macro blocks.

Even in the intercoded macro blocks also there are some situations which will help you in saving the number of bits. you will be finding that there are some parts of the image where there is no motion that is involved but there is some change in the contents in which case what happens is that the motion vector is zero whereas the content has changed, content in that particular area has changed so you can say that you can say that in those areas you will not be sending any motion vector but you will be only encoding the frame to frame difference for those macro blocks.

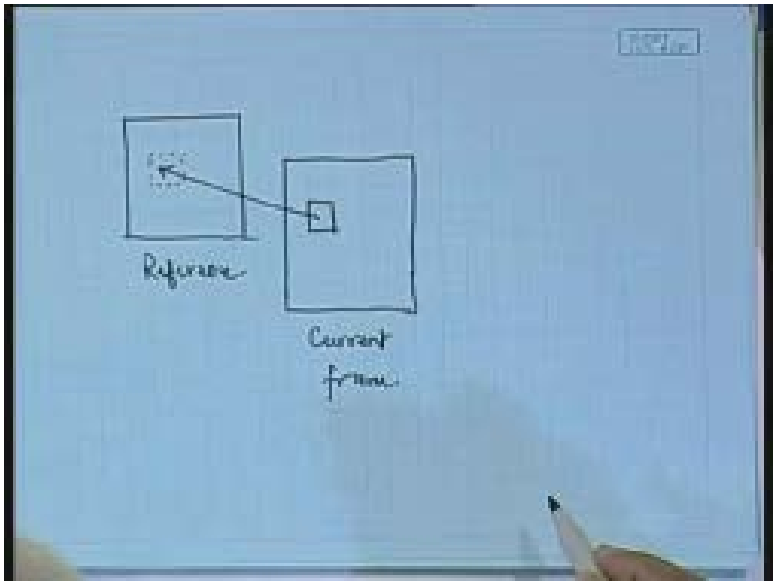
You may observe, like for the background macro blocks, the macro blocks which corresponds to the background region, like say for example in this frame what you are viewing the background information the background is a color that is what we were finding, now in that background information there is no change from one frame to the other. So in those macro blocks that the macro blocks which correspond to those regions it is neither having any change in intensity value nor is it having any motion vector; motion vector for those macro blocks are going to be zero. So best thing what you can do is that you can skip encoding those macro blocks. So there is also a skipped mode that is defined.

Thus, mode of the macro blocks could be inter, intra, skipped macro block and within inter also there is a variation that whether you want the forward encoding to be on, whether you want backward encoding to be on; forward prediction to be on or backward prediction to be on so all these different modalities are possible so essentially this is how the mode of the macro blocks is going to be defined and now with this concept I come to the latest H dot latest video coding standard which is the H dot 264.

Now mind you, in the conventional way of motion estimation how are we representing the motion vectors?

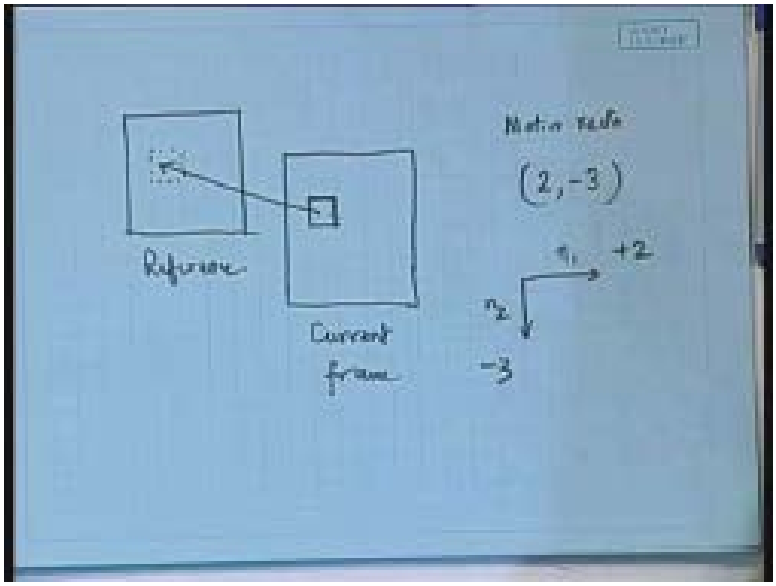
Naturally we are taking one of the we are taking the current frame so we are going to predict the motion corresponding to the macro blocks or the blocks in the current frame with respect to the previous or the reference frame. So there is a current frame and there is a reference frame and for every block we are finding the motion vector by consulting the reference frame.

(Refer Slide Time: 20:30)



Now in such kind of cases whenever we are indicating the motion vector the motion vector will be in terms of pixels; it should be in terms of pixels. If the motion vector is say (2, minus 3) if this is the motion vector that means to say that I take a candidate block and if I apply a displacement of plus 2 units in the n 1 direction, this is the n 1 direction where we apply a displacement of plus 2 unit (Refer Slide Time: 21:14) and this is the n 2 direction where I apply a displacement of minus 3 units that means to say that I go by three units up over here, in that case by applying this motion vector I will be able to find out that which part in the reference image does it correspond to.

(Refer Slide Time: 21:44)



Now these motion vector elements are all integer elements because we are estimating the motion as integer units. Now, think that what could be the problem of having integers? You see, how is the motion observed in the image. It is a real world movement so there is no guarantee that the real world movement will be restricted to the integer movement space. Your pixel resolution is there, you have a certain pixel resolution so in terms of the actual movement it is in the real space so I may be having a displacement of 0.1 units or 0.7 units or 0.77 units like that. So what we are doing is that we are essentially because we are specially sampling the image and we are having some integer grid that is why we are saying that if it is 2.77 then I will be representing it as three units of displacement. If it is 2.03 I will be representing it as two units of displacement.

Now we can achieve some better accuracy if we go in for a subpixel accuracy representation. Now it looks very odd that you may be immediately confused that how am I going to estimate my motion with subpixel accuracy. Let us say for example how am I going to measure my motion vector with plus minus 0.5 pixel accuracy that is to say half pixel accuracy.

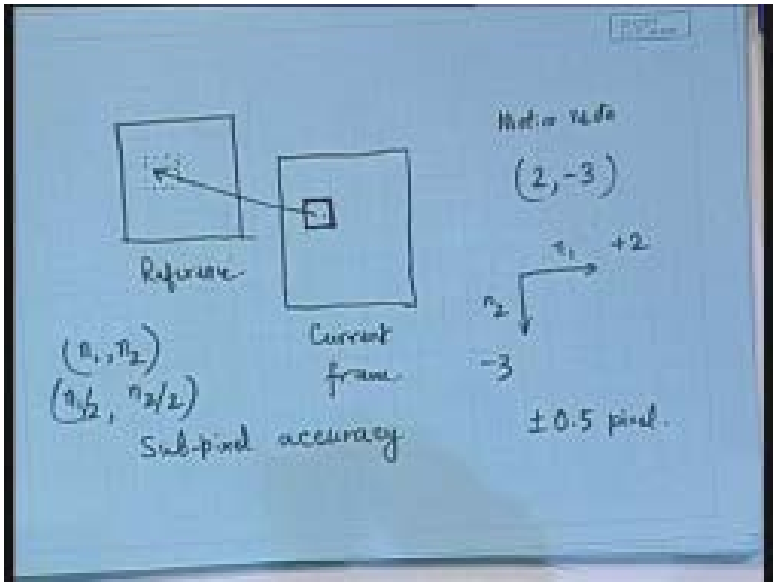
Well, the simple thing what you can do is that you apply interpolation on the image. Whenever you are applying interpolation on the image; there are many techniques to apply interpolation.

One can apply by linear interpolation or one can use some other nonlinear fitting algorithms, there are many such algorithms which are available in the image processing literature. So essentially if you are incorporating some kind of interpolation then you can increase the resolution of the image fictitiously.

Let us say that if you are having N by N image; if your image size is N by N then you can convert this into an image of space 2N by 2N how by interpolating which means to say that if you are having a pixel here, a pixel here, a pixel here like this (Refer Slide Time: 24:46) then it is possible for us to predict the pixels which are there in between. So these are say one unit so here to here it is one unit the unit distance but at 0.5 distance I can apply interpolation and get the intermediate value.

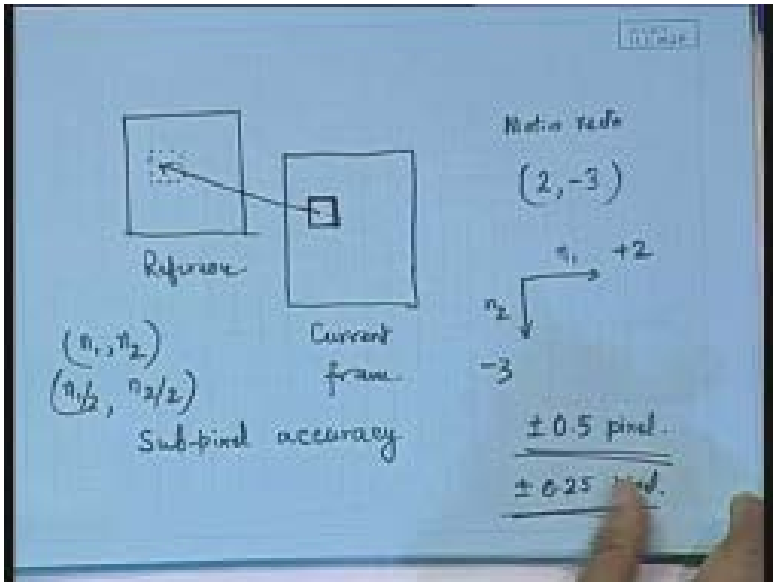
Therefore, all that I am indicating by red these pixels can be interpolated. So you see, original image was a 4 by 4 image but the interpolated image is 8 by 8. In fact, there will be some black pixels even here also so it is..... now increase the resolution of the image has been increased to 2N by 2N. Now if we increase resolution and at the increased resolution we search for the block in the current frame with the increased resolution block in the reference frame in that case the displacement that I will get is in pixel unit but in the enhanced space which means to say that if the motion vector that I obtain is (n 1, n 2) in the increased resolution space then in the original resolution space my motion vector will be (n 1 by 2, n 2 by 2).

(Refer Slide Time: 26:47)



So, if N 1 happens to be 1 then N 1 by 2 happens to be 0.5 so I can I have sub pixel accuracy. This is in this case say half pixel accuracy and now this can be extended to make it quarter pixel accuracy a resolution of plus minus 0.5 pixel which means to say that I have to interpolate to obtain four pixels within two consecutive works. in the earlier case I was having one pixel in between another, in this case totally we must have four pixels so that an N by N image can be enhanced to 4n by 4n image by interpolation and if we carry out the motion vector estimation in this space then we get plus minus 0.25 pixel accuracy which is a quarter pixel motion estimation.

(Refer Slide Time: 27:56)

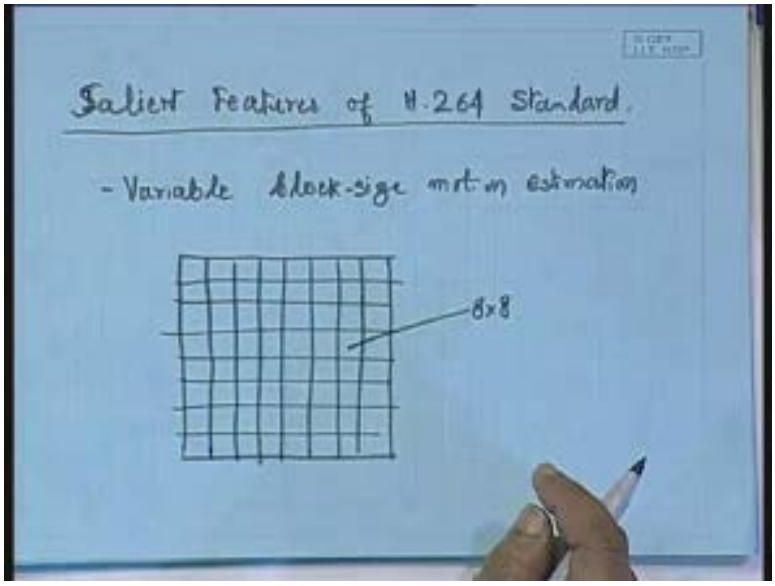


Now let us see what are the salient features that H dot 264 offers to us. Yes, questions? [Conversation between Student and Professor - Not audible ((00:28:02 min))] Yes, very very good question.

Now the motion estimation, I mean, in order to do the interpolation you did not have to do it in all the Y, C r and C B space because after all the displacement is going to be similar in both Y and C r, C B space so only one is enough in fact it is done in the luminance space only because in luminance space if we do the motion estimation then one can represent that to be the motion vector. So luminance is that.

Now the salient features; so let us list out the salient features of H dot 264 standard. The first is that it has variable block size motion estimation. Now this is a very important issue which I need to elaborate. Now you see, so far I was saying that we subdivide the frame into non-overlapping blocks of some fixed size, fixed size means that fixed size could be 8 by 8 so all the fixed sized fixed block sizes it is partitioned to conventionally so this is 8 by 8 and all of these are 8 by 8 block sizes.

(Refer Slide Time: 30:02)

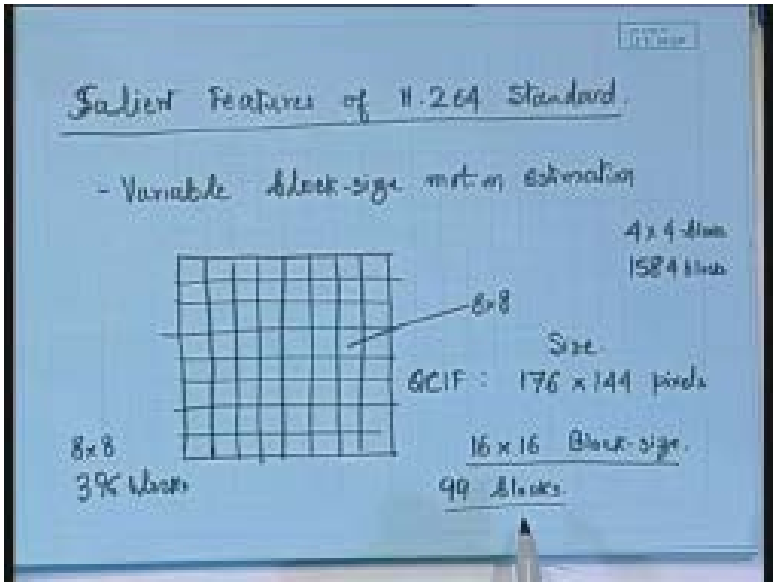


Now let us first discuss that why is this 8 by 8 why not 4 by 4 or why not 16 by 16 or why not 32 by 32; who decides that it is 8 by 8? Well, if we make the block size to be very large what advantage do we have?

First advantage is that, say for example let us let us take some specific example; let us say that the image size is 176 by 144 pixels say. Say we use this size. In fact this is a very standard size which is used in the small sized images in fact what you are observing in your video phone screens is of this size 176 by 144 that is the typical resolution that is what you have.

Now in 176 by 144 pixel size this is referred to as quarter common intermediate format or QCIF is the standard size that is adopted. Now let us say that we have a block size of 16 by 16. now this means to say that so if we use 16 by 16 block size then we are having 176 by 16 11, 144 by 16 that is 9 so we have totally 99 blocks. If on the other hand we make the block size as 8 by 8 we are going to have 396 blocks. If on the other hand I have 4 by 4 blocks then it will be 396 into 4 792 1584 blocks. Now which one is better?

(Refer Slide Time: 32:24)

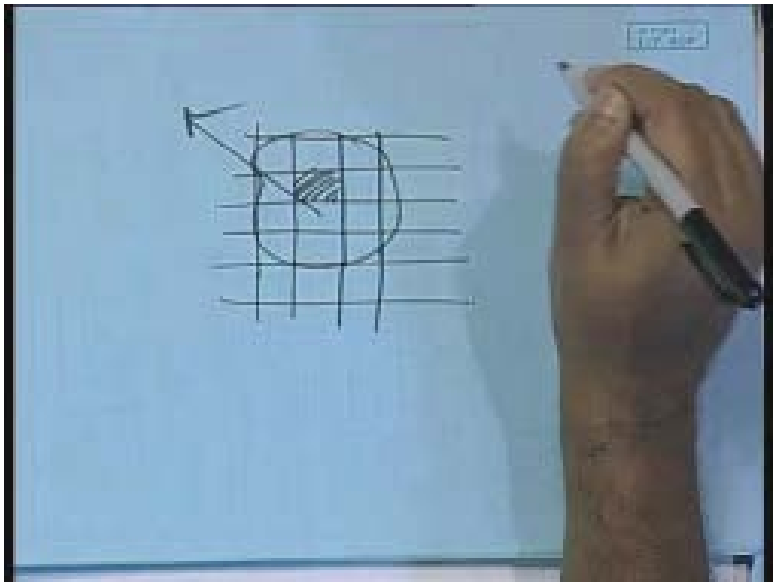


Now if I look at the number of bits consumed in estimating the motion vectors then this is definitely the best because only ninety-nine motion vectors are to be transmitted whereas if I use 4 by 4 block size then 1584 motion vectors are to be transmitted just how many sixteen times of this, sixteen times this those many motion vectors to be transmitted. So naturally encoding the motion vector this will consume much more number of bits so definitely from that point of view larger the block size is better for us. If we could make it 32 by 32, it is still better. But then what is the difficulty if we keep on increasing the block size.

Well, there the difficulty is that whenever you are having a block based motion you are having an inherent assumption that as if to say the whole block is moving with the same displacement. So you are assuming some sort of homogeneity within that block or you are assuming that this particular block is a part of a rigid body which moves as a whole.

Now if that is so then assigning a single motion vector for 16 by 16 blocks makes sense. Like say for example, if I have an object like this a rigid body object and supposing I have subdivided the image into a set of 16 by 16 blocks like this these are all 16 by 16 blocks, now you see this block is fully inside the object.

(Refer Slide Time: 34:34)



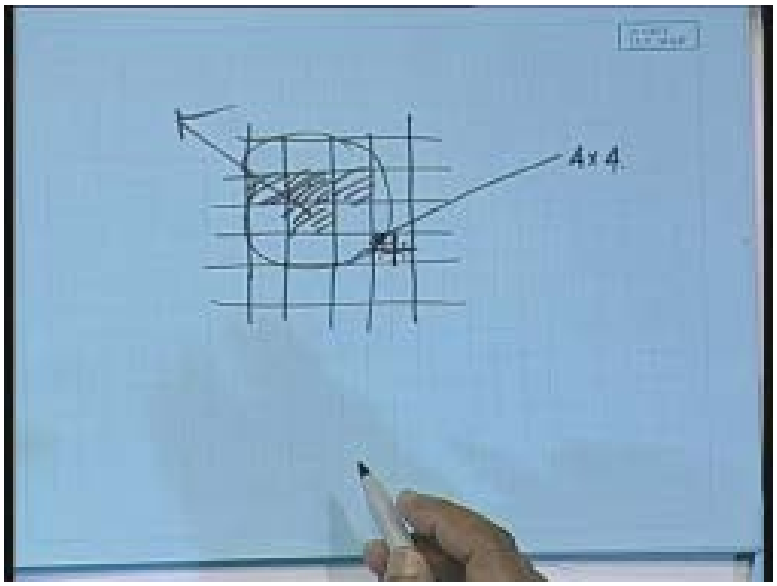
Now, if the object is having a displacement vector like this, this particular block is having that displacement even the next block is also having, even all these blocks which are fully contained but there are some blocks which you are observing which is having only a part of the object covered; like say for example take this block (Refer Slide Time: 34:53) this part of the block is moving but rest of the area is pertaining to the background so it is not moving.

So, if I assign a motion vector corresponding to this object to this entire block I am making an erroneous conclusion; I am imparting a false motion to the background area also. So what is the solution if instead of treating instead of assigning a single motion vector to the entire 16 by 16 block size if I subdivide the block into four 8 by 8 blocks then I can assign motion vectors individually to these four.

Even here also you find that here zero displacement, here zero displacement, here zero displacement but this entire 8 by 8 is going to have the motion vector corresponding to this part which is not even engulfing 8 by 8 area. So we can say that this 8 by 8 area is a moving area, let me sub partition this area further into four sub-blocks of 4 by 4 and by doing that I will be finding that it is only one of these 4 by 4 areas which is moving and rest of the three 4 by 4 areas

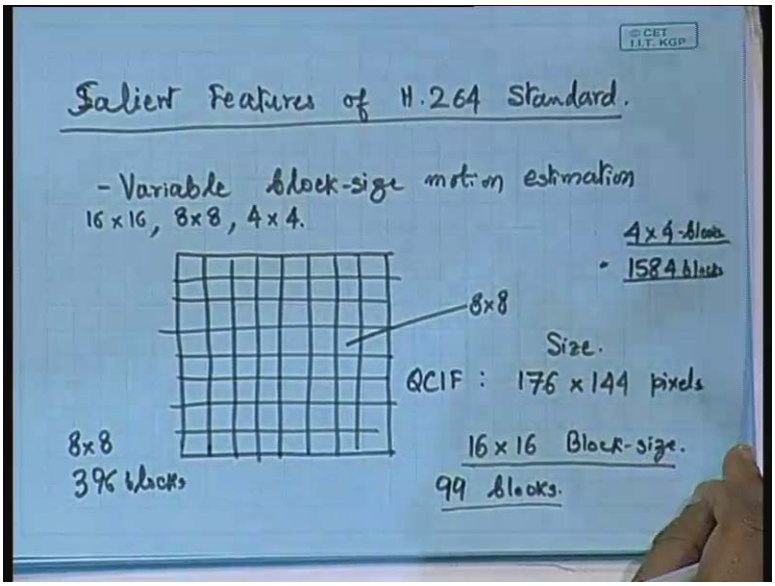
are stationary. So only this particular block will be so this block which is of size 4 by 4 to which a non-zero motion vector is assigned.

(Refer Slide Time: 36:32)



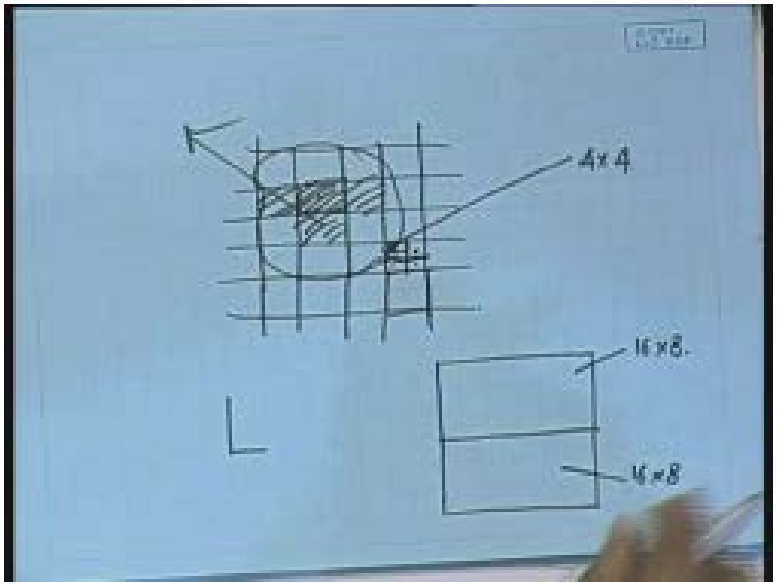
Now, for the rest of the area what I can do is that for this entire 16 by 16 there is no problem I can assign a single motion vector. For this 16 by 16 I can assign a single motion vector that means to say that it depends on the parts of the image. Wherever any block corresponds to the boundary of an object it makes more sense in subdividing that block into further sub-blocks. That is the advantage that one gets as the variable block size motion estimation meaning that for some parts you can keep 16 by 16 block size, for some parts of the image you can subdivide the 16 by 16 blocks into 8 by 8, for some parts you can subdivide into 4 by 4 this 8 by 8 could be sub partitioned into 4 by 4 so it is a variable block size motion estimation.

(Refer Slide Time: 37:35)



Not only that, the H dot 264 standard has got so much of flexibility that even whenever you are having a 16 by 16 block size let us say that this is a 16 by 16 block size I can not only subdivide this into four sub-blocks of 8 by 8 and have four motion vectors; I can also say that for this 16 by 8 there is one motion vector, for this 16 by 8 there is another motion vector (Refer Slide Time: 38:10) so 16 by 16 can be portioned into four different ways.

(Refer Slide Time: 38:11)

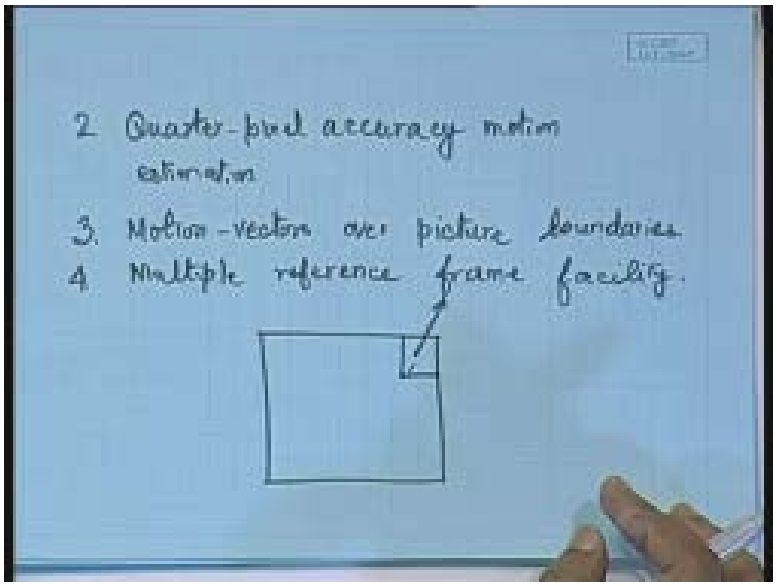


If this is the 16 by 16 block size say this is 16 by 16 then it is possible for us to have a vertical partitioning and have..... this we are going to call as 216 by 8 (Refer Slide Time: 38:31) then we can have 8 by 16 by having horizontal partitioning, one can have four 8 by 8 partitions so this is four 8 by 8 partitions and similarly every 8 by 8 block can be partitioned into two 8 by 4 blocks or two 4 by 8 blocks or four 4 by 4 blocks. This is the variable block size motion estimation. So this is one advantageous feature which was not there in the earlier versions of the standards, it is there in eight H dot 264.

Then there is another aspect and that is the quarter pixel accuracy motion estimation. So the second is quarter pixel accuracy motion estimation the second feature of H dot 264, this I already mentioned. The third point is motion-vectors over picture boundaries. Now what I mean to say by this is that say this is the candidate frame (Refer Slide Time: 40:26) and we have some parts of the object, now here the motion vector could point outside the boundary also; means what happens is that this may be viewable in the part of the space..... I mean, in the reference frame wherein the viewing area of the reference frame you cannot find the object but actually in the memory whenever you have stored the reference frame then part of that is still available in the

memory but not in the actual viewing area so you can point out to those areas in the motion vector.

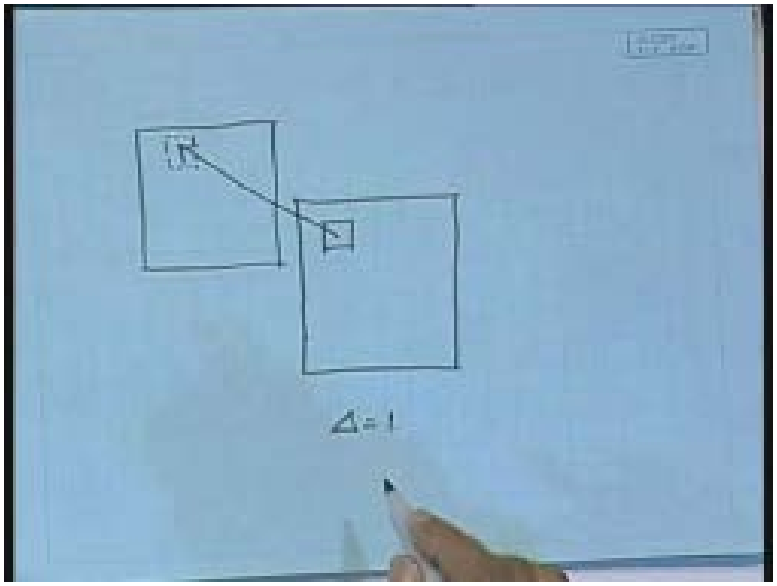
(Refer Slide Time: 41:46)



The next feature is multiple reference picture motion compensation multiple reference frame multiple reference frame facility. Let me tell you that what is the multiple reference frame facility. See, we assume that we are just predicting the motion with respect to the previous frame or whenever we are doing a bidirectional prediction there we are predicting with reference to an already predicated frame. So, whenever we are having B frames there we are predicting forward I frames or P frames and the backward prediction we are making from some P frames like this we are doing. But the frame references have to be fixed throughout the image.

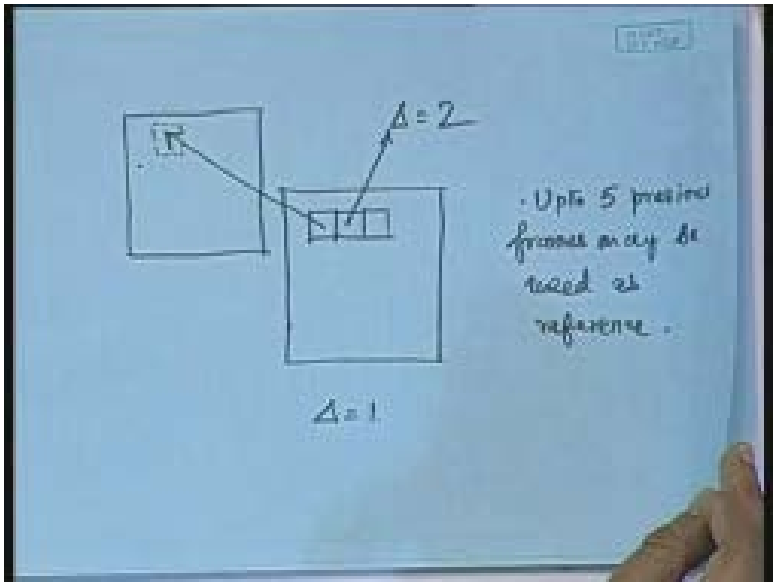
What I mean to say is that if this is a frame if this is the candidate frame and we are having motion estimation with respect to the immediate past frame then what we are using is delta equal to 1; delta equal to 1 means a delay of only one frame that means to say that there every candidate block is searched with respect to the previous frame so delta is equal to 1.

(Refer Slide Time: 42:59)



Now in the case of H dot 264 what happens is that some of the macro blocks you can have a reference of immediate past frame as delta equal to 1 but for some frames you can define some different delta also. Like say for example; if I define this to be the delta equal to 1 and I define the next macro block with delta is equal to 2 what it means is that for this macro block the reference frame is not this one but what I have two frames back. So there is a provision to have up to five previous reference frames; so up to five previous frames can be accommodated and can be used as reference may be used as reference. So this is the multiple reference frame facility.

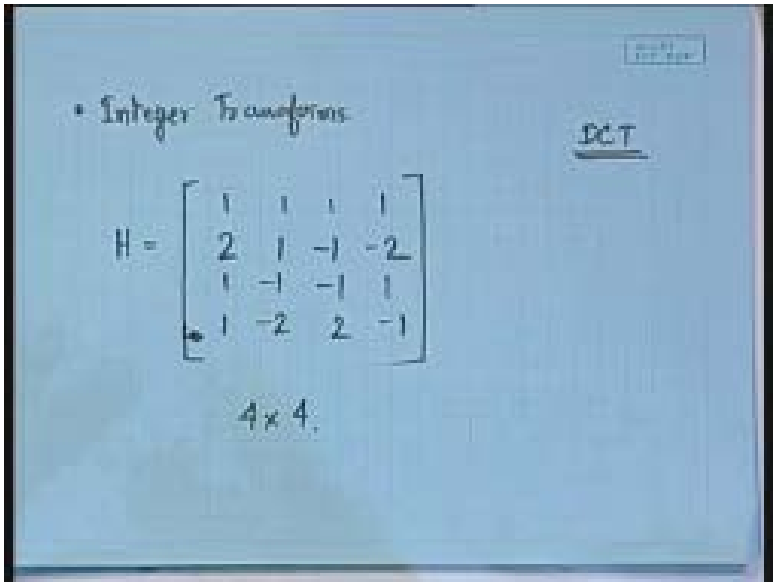
(Refer Slide Time: 00:44:07)



Then another point where it has gained is the transform. The transform is actually an integer transform. In all the previous standards they have used the DCT. Now the DCT coefficients as you know that all the DCT kernel values they are basically the cosine values. The cosine values are real quantities which are represented within value within a range of minus 1 to plus 1 real values. But in the case of the H dot 264 instead of using the real value DCT as the transform coefficient a new transform which is referred to as the integer transform is defined and integer transforms kernel is like this. So its transformation kernel..... actually the integer transform is applied on 4 by 4 blocks not on 8 by 8. in the earlier standards 8 by 8 was followed.

As I mentioned, in the JPEG also and even in the standards like the MPEG-1 MPEG-2 everywhere the block size is treated as 8 by 8 but in H dot 264 the integer transform is done like this that it is a transformation kernel having these entries. So this is the integer transform array, 4 by 4 integer transform array.

(Refer Slide Time: 46:16)



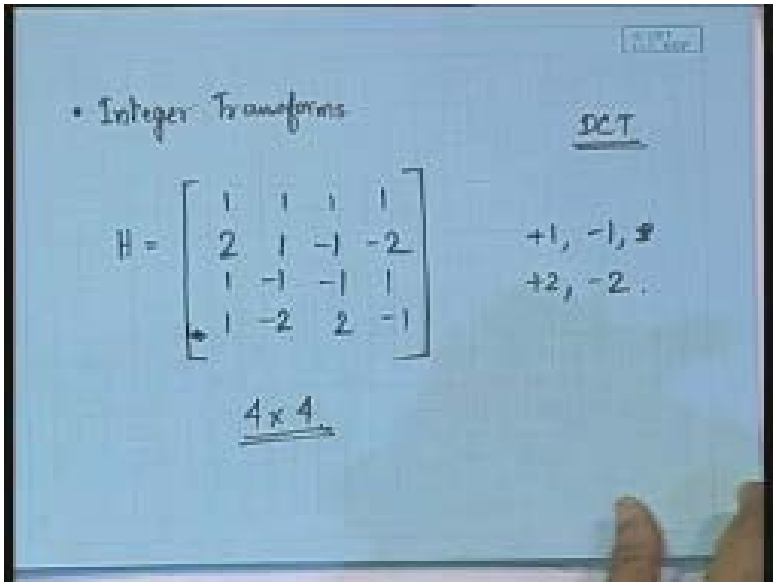
Now what do you observe in these elements?

The elements are either plus 1 or minus 1 or plus 2 or minus 2. So the elements can assume only these four values: plus 1 minus 1 plus 2 minus 2.

What is the advantage?

Advantage is in the hardware implementation. You see multiplication by plus 1 is a trivial multiplication so whatever value is there you are retaining the same value. Multiplication by minus 1 means it is complementing the sign; it is just taking the 1's complement of that. Multiplying by 2 means that you are making shift by one position so hardware-wise it is very easy to do and multiplication by minus 2 means you are doing shift but doing a negation so take the 2's complement and do a shift so that is why the integer transform can be very easily implemented in hardware and they are using 4 by 4 block sizes. So the hardware unit is very much simplified; the integer transformation computation is not involved at all which is the case in the case of the earlier standards.

(Refer Slide Time: 47:51)



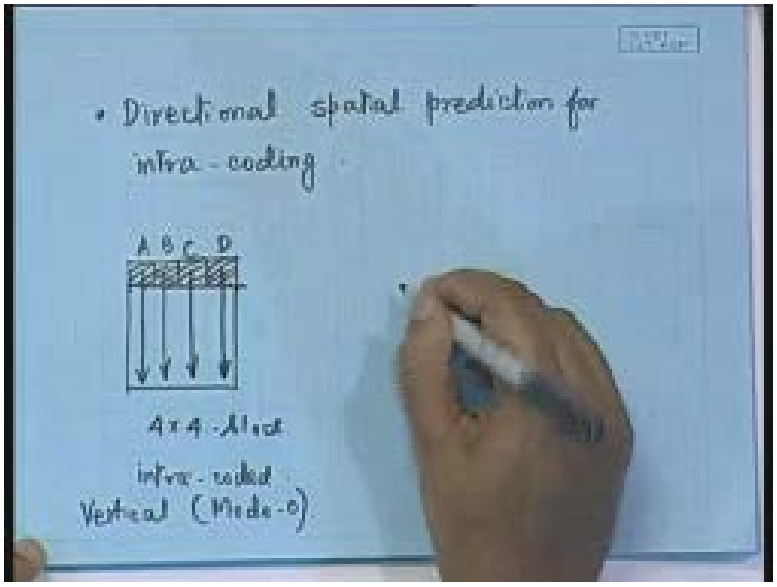
Then another great advantage of H dot 264 is its directional spatial prediction for intra-coding. What we mean by directional spatial prediction for intra-coding is like this that you see, in the intra-coded blocks or intra-coded macro blocks what we do is that there is no prediction that is employed no past reference can be used for the intra blocks or intra macro blocks. But in the earlier standards that is why the intra blocks where for intra-coding the blocks where just treated as intra-coded blocks and there was no prediction that was employed so directly whatever intensity values are there based on that the transformation was obtained and encoded.

Now, it is seen that many a times the blocks which are to be encoded in the intra-coding bares some very close similarity with the neighborhood blocks. The neighborhood blocks which are appearing above or the neighborhood blocks which are appearing to the side or the neighborhood blocks which are appearing on the top left diagonal these blocks could be used for predicting the coefficients of the inter-coded blocks. So it is done something like this that supposing we just have a 4 by 4 block like this so supposing this is a 4 by 4 block which is to be intra predicted which is to be intra-coded

now one of the way whereby we can do is that we can take the four pixels which are there immediately on top of this block this pixel say A the pixel B which lies just above this the pixel C which lies just above this and the pixel D.

Now I can use these four pixels A B C and D to predict this intra-coded block by simply copying this A's intensity values into all these four positions, by coping B into all these four positions, by coping C into all these four, by coping D into all these four. So in this case this is a vertical prediction. So we are using the top pixel to predict this 4 by 4 block and this is vertical prediction and vertical prediction is called as the mode-0 mode-0 prediction.

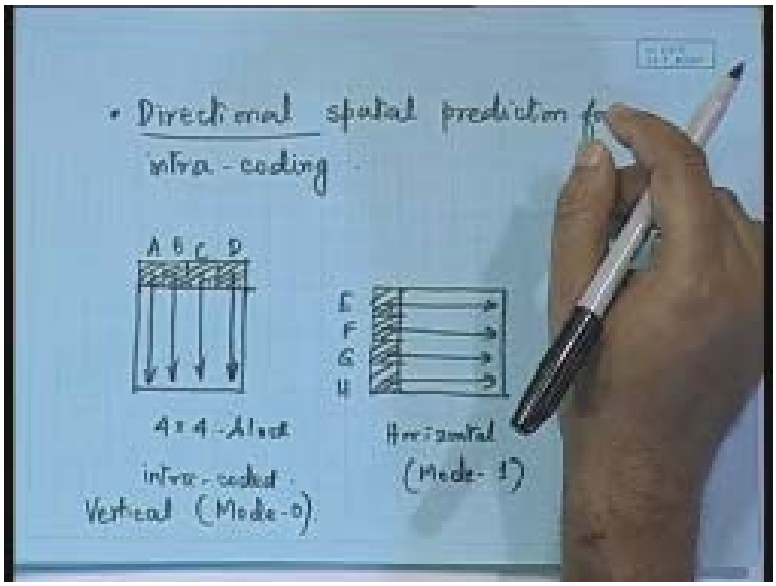
(Refer Slide Time: 51:28)



So, just like the way one can do a mode-0 prediction one can do a horizontal prediction also, how? let us take four pixels on the side, let us take the pixels E F G and H E F G H and we use the value of E in all these four pixels, use F in all these four, G in all these four, H in all these four so in this case it is a horizontal prediction that is what we are doing and this will be referred to as mode-1 prediction

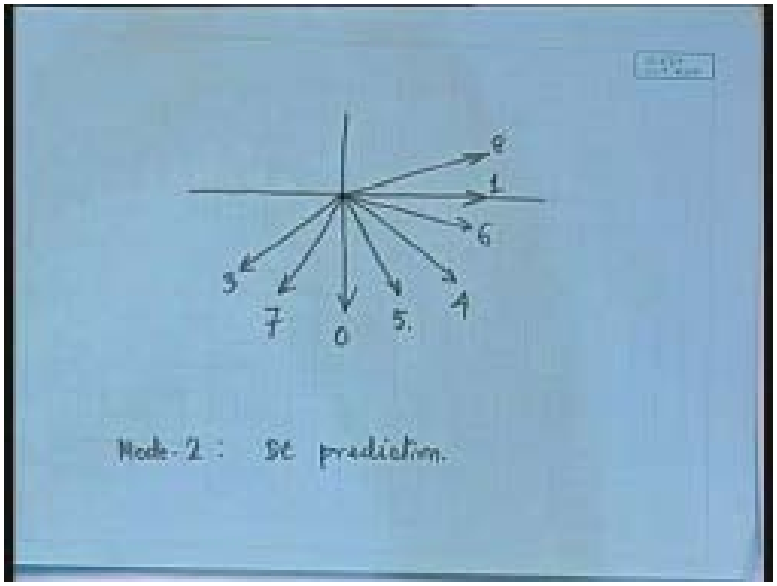
In fact there are eight different directions that are employed in the directional prediction of H dot 264 and the eight different directions are like this that this vertical one we have we are calling as the mode-0, the horizontal we are calling as mode-1.

(Refer Slide Time: 52:17)



Now mode-2 is a very interesting one. Mode-2 is actually what is known as the DC prediction. DC prediction means that the average intensity value is used to predict the next frame. So it is not a directional prediction we have to predict the block so it is not used in the direction but the direction 3 is used in this. So direction 3 is this one propagating 45 degree (Refer Slide Time: 53:20), direction 4 is this direction that means to say that predicting from this one so this is direction 4, direction 5 in between direction 0 and 4, direction 6 is in between 4 and 1; all these things are 22 and a 1/2 degree apart, direction 7 is in between 0 and 3 and direction 8 is this one 22 and a 1/2 degree above this so totally 4 5 6 7 8 so eight different directions 0 to 8 mode-2 is used as the DC prediction but mode 0 to mode 8 are used similarly for predicting from the different directions.

(Refer Slide Time: 54:25)



So just like the way I talked about mode-0 like this, mode-1 like this, if you look at mode-4 then mode-4 will be a diagonal propagation; mode-5 will be with a 22 and a 1/2degree means again it will get propagated in some manner like this that not all the pixels will be diagonally propagated, it will be sampled and made into 22 and a ½ degree directions. So these are some of the salient features of H dot 264. Next class I will summarize the coding technique of H dot 264 along with one or two examples, thank you.