Digital Voice and Picture Communication

Prof. S. Sengupta

Department of Electronics and Communication Engineering

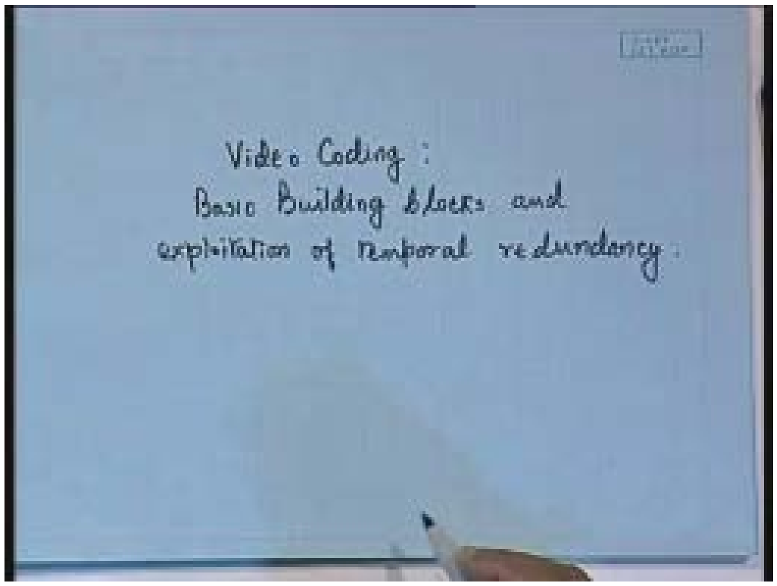
Indian Institute of Technology, Kharagpur

Lecture - 23

Video Coding: Basic Building Blocks

This lecture is on the video coding and in this lecture we are going to discuss about the basic building blocks and how to exploit the temporal redundancy.

(Refer Slide Time: 1:28)



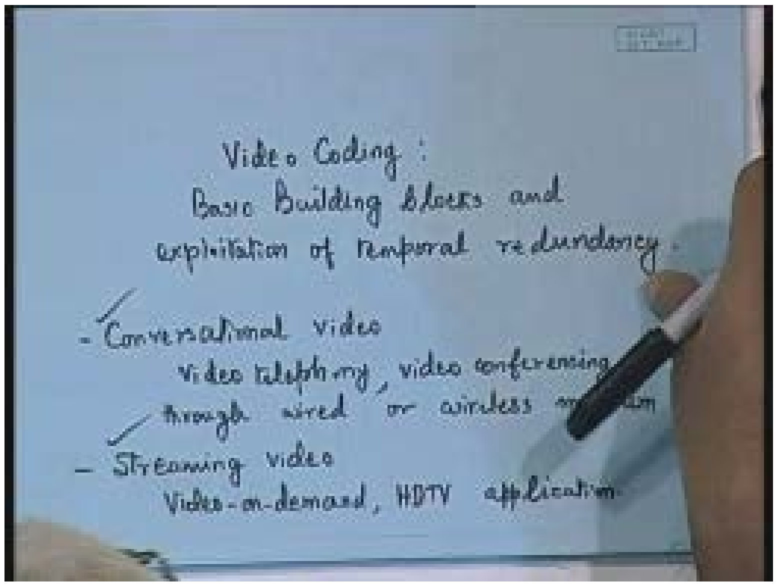
Video Coding :
Basic Building blocks and
exploitation of temporal redundancy.

So far we have seen the image compression standards. Now a still image as you have already seen that a still image does not have temporal information it has got only the spatial information and once for all we have to send a frame of image and that is it so that should be encoded by the encoder at the transmitter end and at the receiver end it should be decoded and the receiver can reconstruct the image whereas in the case of video we have a continuous stream of images which will be available from a video camera or some stored video medium.

Now what are the different applications?
The applications of video communications one can broadly categorize into two major application areas: one is what is called as the conversational video and under this conversational video category we have video telephony, video conferencing. Video telephony and video conferencing these could be through wired or wireless media and the other application major application domain is the streaming video like this. This video which you are watching through the net and you are able to download it into your own terminal and able to see as a video on demand this is a streaming video application.

Therefore, video on demand is one of the major application of streaming video where some recorded video is already there; recorded video in the digital medium, in the computer or in the digital medium it is there and you can just get that video stream and play it into your system. That is what you are having as a streaming video. Streaming video means that as the video stream is being sent you are also simultaneously watching, you need not have to wait until the complete video is downloaded. Even if a part of it has been sent from the encoder to the receiver, once you start receiving, once you are in a position to play some frames, the playing can commence while the subsequent video stream would be sent from the encoder to the decoder so that is the streaming application. So one has got video on demand, HDTV applications then image or video database services etc those are the streaming video applications. So these are the two major areas conversational and streaming video which will be addressed in the video coding.

(Refer Slide Time: 5:33)



Video Coding :
Basic Building blocks and
exploitation of temporal redundancy.

- Conversational video
  video telephony, video conferencing
  through wired or wireless medium
- Streaming video
  Video-on-demand, HDTV application

Now, what is the basic difference that we are going to have between the still image coding or compression and video coding?

The first and foremost difference is that, in the case of video we are having the temporal aspect of the signal. The signal is not only varying in space but the signal is also varying in time. And there is a great deal of similarity between the signal that varies in time. Because what is being done is that the video is captured at a particular frame rate. let us say that the frame rate of the video is 30 frames per second, so 30 frames per second means within one second we are capturing 30 different frames; at a time of 33 milliseconds apart we are just sampling each and every frame and if we are comparing successive frames then we will be finding that within the successive frames there will be a great deal of similarity because the physical movement that can take place within a time span of 30 milliseconds is highly limited. I mean, even this video what you are watching here how much of movement can you see from one frame to the next?

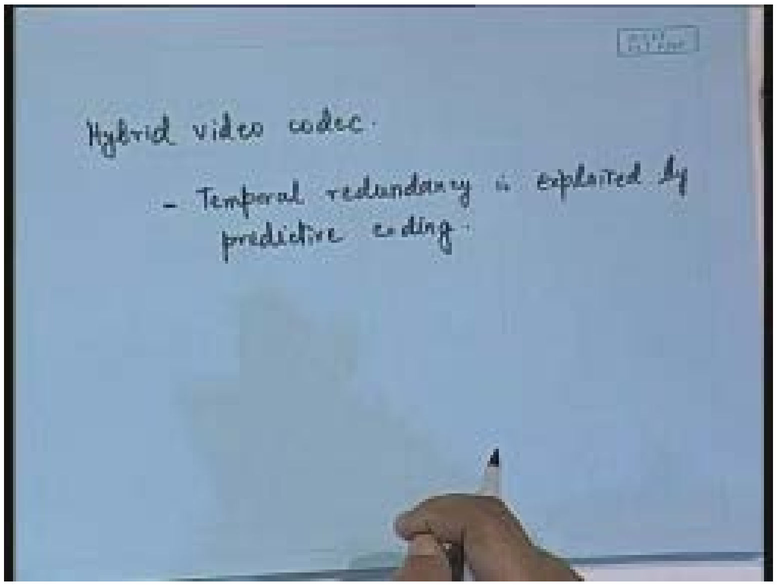
If you are really analyzing you will be finding that the background is stationary, it is only that when a person is talking like here I am talking you will be finding that there is a lip movement, there is eye movement, sometimes shoulder movements are there so a limited degree of movement you will be observing so that there is a great deal of temporal redundancy that is present and this temporal redundancy also can be exploited in addition to the spatial redundancy.

In the case of steel image we could see that only the spatial redundancy could be exploited. And how did we exploit that? We had transformed the steel image from its original spatial domain into a transformed domain space like the discrete cosine transform or the discrete wavelet transform and then we worked with those transformed coefficients and encoded and sent the bits sent the bit-stream. We had essentially use the transform domain approach; although there are alternatives to the transform domain approach also like just as in the case of the speech we could use the differential pulse code modulation scheme, adaptive differential pulse code modulation schemes etc........ images also can very well make use of that; DPCM could be used for images also.

What is DPCM or an ADPCM whatever you can talk of? It is basically a predictive coding technique. But that prediction has to be a spatial prediction. In the case of still image one has to predict a pixel by observing the values of the past transmitted pixels. In the case of speech what is the mechanism of prediction? We had temporal prediction that the time domain samples which are arriving based on the past time domain samples we are predicting what the next sample is going to be, so it is a temporal prediction whereas in still image it has to be a spatial prediction. But now in the case of video we are having both variations with respect to space and variation with respect to time.

As far as the spatial redundancy is concerned it is a proven fact that transform domain technique can exploit the spatial redundancies much better as compared to predictive encoding for spatial domain. But again looking at what we had learnt in the speech, we had seen that for time domain redundancy exploitation predictive coding is also a very good and effective technique. So why not have a combination of these two techniques? As far as the removal of the spatial redundancy is concerned adopt a spatial domain technique and as far as removal of time domain is concerned adopt a.......... I mean adopt a spatial domain adopt a transform domain technique whereas in the case of removal of temporal redundancy there you adopt a predictive coding technique. So what essentially evolves is a hybrid coding technique. So it leads to a hybrid video codec hybrid video codec where the first and foremost thing what is done is that the temporal redundancy temporal redundancy is exploited by predictive coding.

(Refer Slide Time: 11:38)



Hybrid video codec.

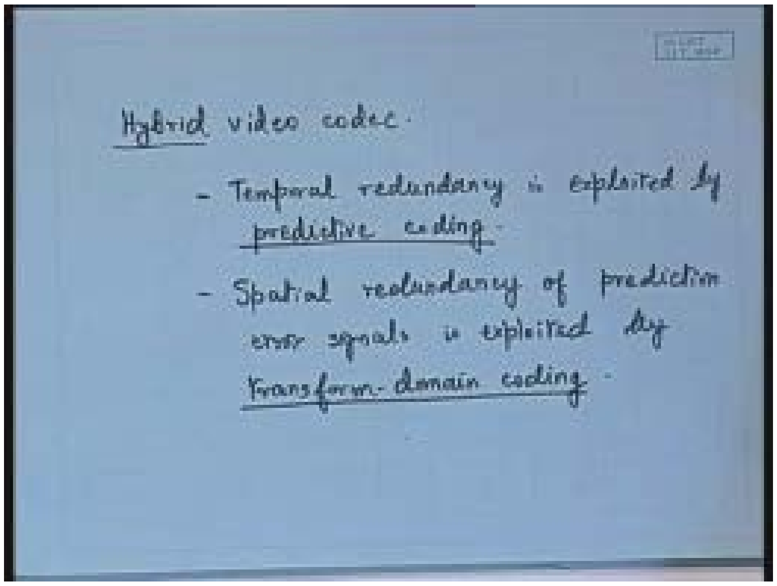
- Temporal redundancy is exploited by predictive coding.

Now how would you perform a predictive coding?

If you are accepting the video sequence then you are also having successive frames are coming and if you have a storage if you have a memory to store the past frame then you can make use of the past frame to predict what the present frame is going to be. Because as I had told you that between successive frames there may not be much of change in the information so the past frame can very well predict what the next frame is going to be and then if you then but but the prediction is not going to be exact definitely because there will be some changes. Even though there are parts where there will be no change like the background the stationary background especially but there will be areas where there will be changes. So if you have predicting only based on the past frame you are going to have some prediction error. That prediction error would have some kind of a spatial similarity. Means the prediction error that we have; I mean, between successive pixels in space we will be finding that there is not much of change in the prediction error signal and that can be very well exploited by a transform domain technique.

Hence, spatial redundancy I can tell it like this that spatial redundancy of prediction error signal the temporal prediction error signals that can be that is exploited by transform domain coding. So, in the video encoding process that we are going to utilize we are going to have a combination of predictive coding for temporal redundancy and transform domain technique transform domain coding for residual spatial redundancy. We can call it as a residue, why residue is because it is the temporal prediction error that we are encoding through the transform domain. So it is a combination of predictive and transform domain coding and that is why this is called as the hybrid video coding.

(Refer Slide Time: 14:41)

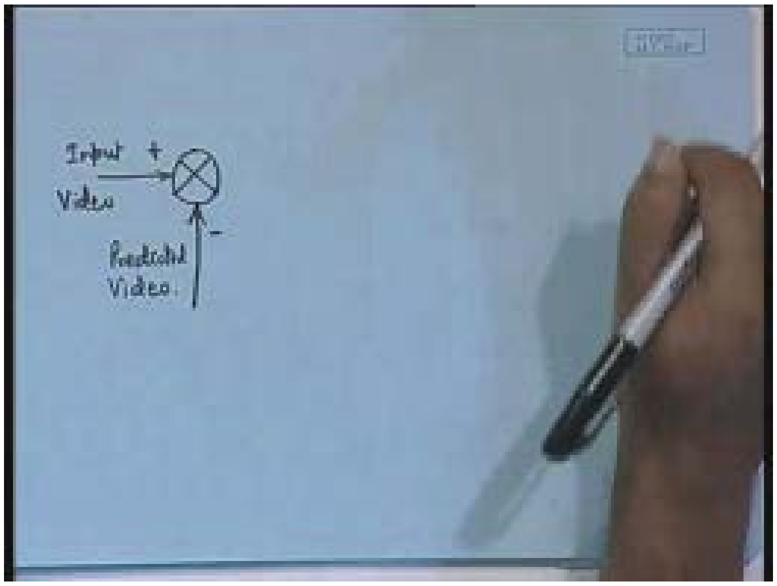


Hybrid video codec.

- Temporal redundancy is exploited by predictive coding.
- Spatial redundancy of prediction error signals is exploited by transform-domain coding.

Therefore, in a typical hybrid video codec what we are going to have is like this that here we are having input video, here we are going to have a block where the input video will be subtracted from the predicted video.

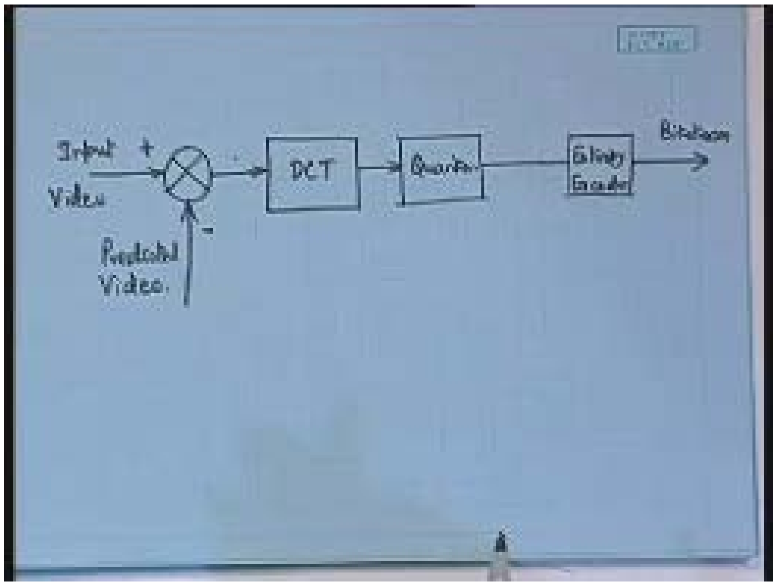
(Refer Slide Time: 15:16)



Now how that predicted video is generated that part we will be seeing. But at least one thing is very clear that if this is with a plus sign and this is with a minus sign in that case what we get here is the prediction error and as we said the prediction error or the residual signal and the residual signal has to be transform codec. So there will be a transform block that transform block would be DCT or DWT, let us say DCT because again DCT is a coding technique that has been used in all the popular video coding standards that has that have evolved through late.

I will be mentioning about the video coding standards very shortly. But DCT is just a specific case of the transform that one can use for the residual signal and this DCT has to be followed up by a quantizer. This blocks we also had remember in the JPEG encoding scheme (Refer Slide Time: 16:27). There it was an original image which was DCT encoder and then quantized, in this case it is the residual video signal. So, the quantized signal then could be coded through the entropy coding technique just like the way we did for the still image so this is an entropy encoder and the entropy encoder will generate the final bit-stream.

(Refer Slide Time: 17:00)



Now this is the forward path of the encoder which looks very clear. That means to say that what this bit-stream is encoding is not the signal but it is encoding the residual signal the prediction error signal. So somehow that prediction error should be that information and how to predict, that information should be conveyed from the encoder to the decoder also so the decoder can in turn generate what the encoder I mean what the original video was. Of course exact reconstruction will not be possible because we have a quantizer present here; quantizer will be a lossy so exact reconstruction will not be there. But you can start thinking about what you will be finding in the decoder because decoder will have a reversal of this process.

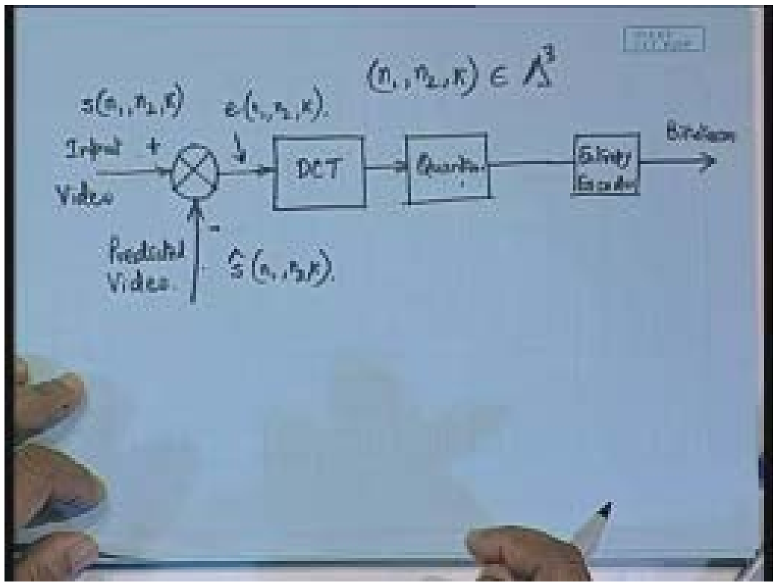
So what will a decoder block diagram look like?
The bit-stream will be the received, bit-stream will be received; the bit-stream will go through an entropy decoder then there is a quantizer in the encoder so we have to have a de-quantizer. What is the de-quantizer? in the process of quantization what are we doing; we are dividing the coefficient value by some quantization matrix elements. In this case these already quantized values have to be multiplied by the quantization matrix element so there will be de-quantizer. And here we had the DCT or the direct transform (Refer Slide Time: 19:14) in this case we are

going to have an inverse transform so we are going to have the inverse transform or in this specific case IDCT inverse DCT we have to do.

But here what do we get?
Here (Refer Slide Time: 19:37) we are going to get only the error signal. But the error signal is not the one that we want to show on to the screen on to the display. In the display we want to show the proper video not the residual signal. But how do we obtain the error signal? It is input minus predicted that is error. Thus, if we add the predicted to the error what do we get, we get input; not the exact input because even this. So if we call this signal as s (n 1, n 2, k) so I have now introduced so s (n 1, n 2) I was so long indicating as the pixel intensity at the position (n 1, n 2); now I am denoting the intensity as s of (n 1, n 2, k) so there are three variables that have come in and what is the third variable the third variable is the time.

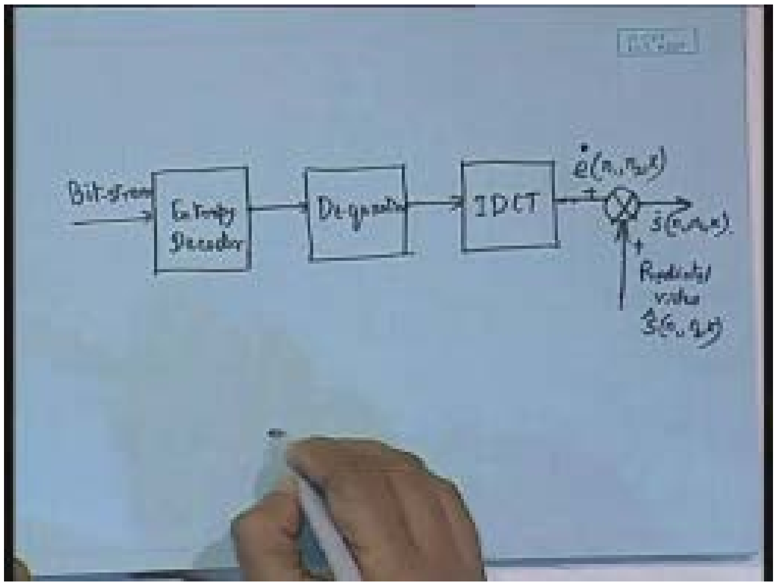
So it is a frame number so this also this is also an integer. So (n 1, n 2, k) all of these are in integer so that essentially I can say that this (n 1, n 2, k) this belongs to a three dimensional integer space lambda cube. So what it means to say is that this is the input video signal and the predicted video signal also will be we can denote as s cap of (n 1, n 2, k) so that the error we will be denoting as e (n 1, n 2, k).

(Refer Slide Time: 21:43)



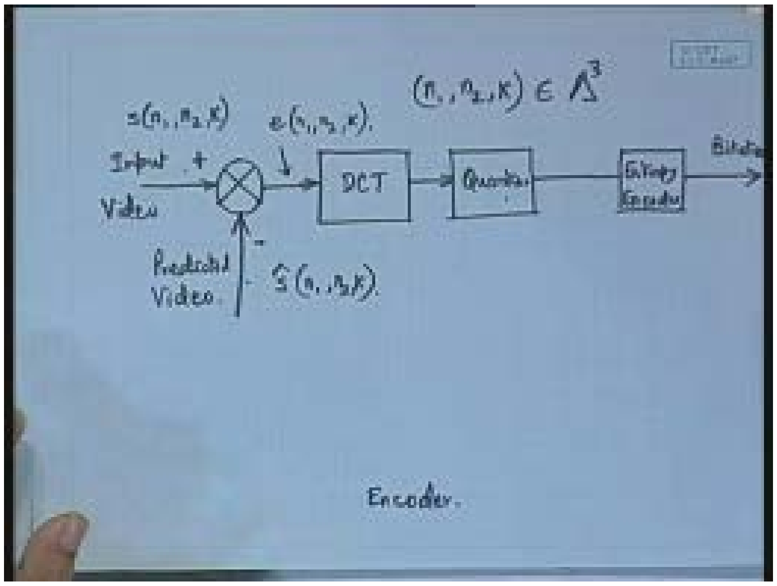
Now here what we are going to get (Refer Slide Time: 21:46); at the IDCT output we want to get e (n 1, n 2, k) but we will not get exactly e (n 1, n 2, k) we will get e dot (n 1, n 2, k) why; because of this quantizer at the encoder. So we have got e dot (n 1, n 2, k) and if we have a block where we add or sum up e dot (n 1, n 2, k) with the predicted video that is s cap (n 1, n 2, k) what are we going to get; we are going to get the reconstructed video or we will write it as s dot (n 1, n 2, k).

(Refer Slide Time: 22:39)



So this will be what we are going to expect at the decoder and this is what we are expecting at the encoder but I have not completed my job yet.

(Refer Slide Time: 22:56)



What I have shown here is only a part of the encoder and a part of the decoder only looking at the forward path. But I have left this issue open that how to generate the predicted video. The only fact that I made a brief mention is that the predicted video should be generated from the past video frames. So I must be having the signal corresponding to s (n 1, n 2, k minus 1) or in general k minus L where L is a number which is greater than zero. So any frame which is s (n 1, n 2, k minus L) with L greater than 0 basically refers to the past frames and if I get some of the past frames then I should be able to predict this one. So how to generate the past frame. Again here I have got only the error signal and that too at this point (Refer Slide Time: 24:05) it is transformed already.

Now you can well argue that, well, generation of the predicted video is not a problem. You take the input video, the input video you take directly, you store it into some memory, you get the past video frame and you compare the past video frame with the input video find out that how much of displacements every block of the image has undergone and then you can generate the predicted video from here.

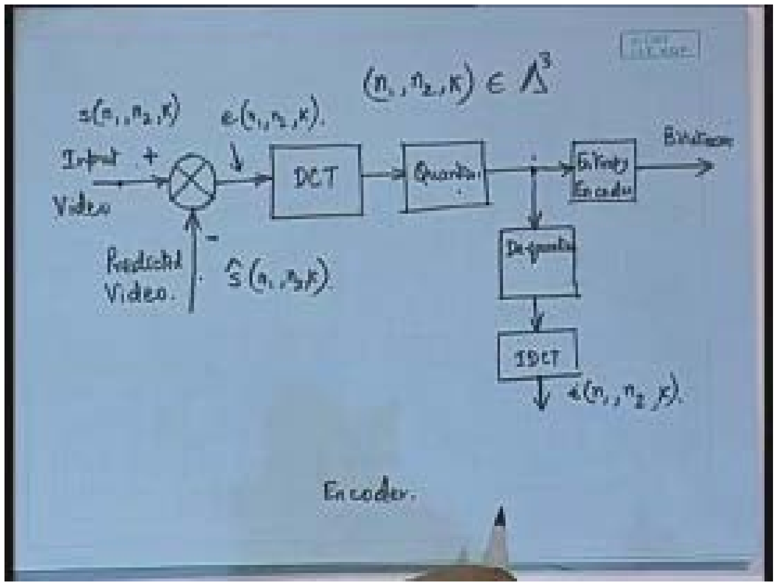
Why do you have to derive the signal from here; because after all if you have to derive you have to derive from this point from the quantizer. Well, encoder can do that. But can a decoder do that? No, because the decoder does not have this input video information anymore. What the decoder is going to have is only this encoded information after the entropy so by a process of entropy decoding because entropy encoding is a lossless process at the end of entropy decoding the decoder gets back exactly what you have over here the decoder will get back at its end.

so the best point should be the So the philosophy of generating the predicted video should be the way decoder is going to predict in the same manner the encoder is also going to predict so that whatever error encoder has got that error residues can be encoded and sent into the bit-stream and the decoder will generate that in a very similar manner because the decoder is also having the presence of this signal but the decoder is not having the presence of this signal so because of this reason we have to tap and derive the stored video from this part.

So how do we do that?

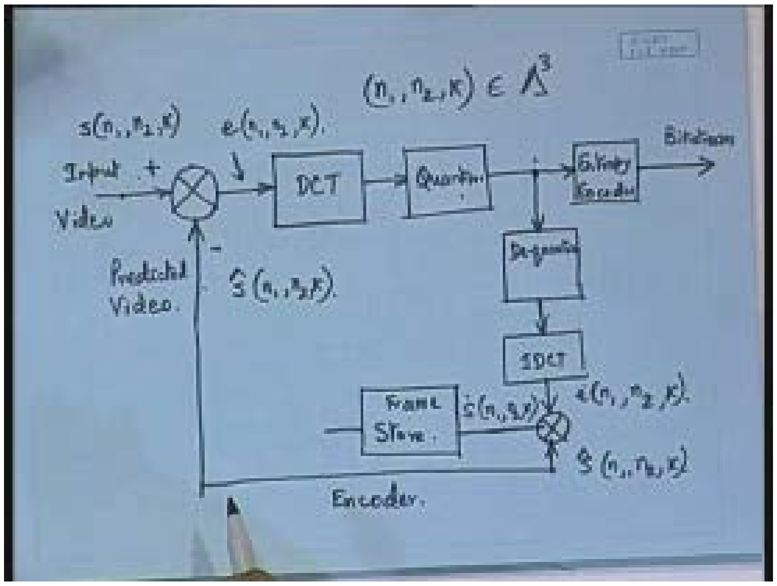
Here we have to do a mimicking of the decoder part. What did we do at the decoder; we had put forward an entropy decoder, de-quantizer, IDCT and generated e dot. In this case what we have to do is we do not have to use entropy decoder because this is before the entropy encoding where we are tapping. So what should be the block that we are going to have over here; de-quantizer. So here we keep a de-quantizer, the next block should be an IDCT and output of IDCT would give me e dot (n 1, n 2, k).

(Refer Slide Time: 27:18)



Now if I have a predicted video; I still do not know how have i generated predicted video; if I have a predicted video in that case what I should do is that I should add up this this is s cap (n 1, n 2, k) that we should add up and what do we obtain we obtain s dot (n 1, n 2, k) exactly what the decoder also is able to obtain provided there is a good mechanism of getting the predicted video. So here we will be getting s dot (n 1, n 2, k) and now we can have a frame store. This is frame storage. So at the output of the frame store what we get we get s dot (n 1, n 2, k minus 1). If it is storage of only one frame then we have s dot (n 1, n 2, k minus 1) available at the output of the frame store.

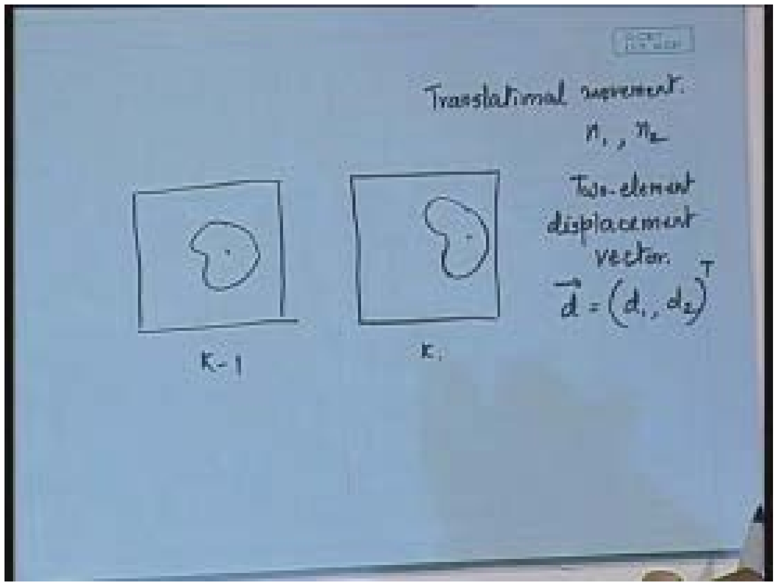
(Refer Slide Time: 28:45)



Now you have s (n 1, n 2, k) the incoming video over here and then you are having the s dot (n 1, n 2, k minus 1) the past frame; you compare the present frame with the past frame and using that what you can get; you can you can find out that there will be some regions where there will be no change in the intensity and there will be some regions where there will be some change in the intensity because of the movements.

Basically it is something like this that you have a frame and supposing you have got an object which is at this position (Refer Slide Time: 29:38) say the centroid of the object is some point (n 1, n 2) so this is at the frame k or this is at the frame k minus 1. And at frame k what will happen is that the same object moves; the very simplistic assumption is that it undergoes a translational movement that is a simplistic assumption. If it does a translational movement in that case what we are going to have; translational movement is going to have an x component and a y component or since we assume the directions as (n 1, n 2) the translational movement will be having one component along n 1 direction and one component along n 2 direction so that the translational movement can be specified as a two element vector. So the translational movement can be specified by a two element vector and we are going to call that as two element displacement vector and we can denote it as d vector where d vector’s elements are (d 1, d 2) transpose.

(Refer Slide Time: 31:15)



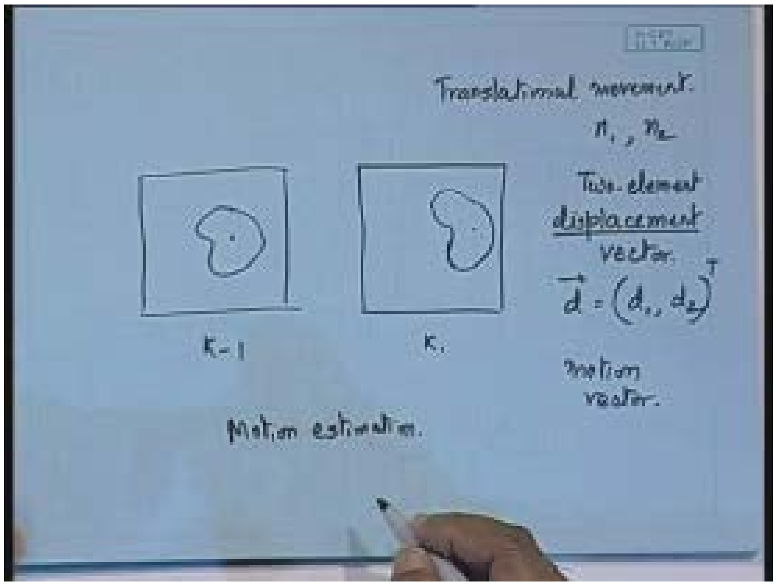
So d 1 is the displacement along the n 1 direction; d 2 is displacement along n 2 direction and this is the displacement vector or also known as the motion vector. Basically the motion vector refers to the same displacement vector only. Now, if we compare this k 1 frame with k then what we will be observing is that there will be some regions where the displacement is zero and there will be some moving objects where the displacements will be nonzero. So we have to estimate those displacements. How we can do is by matching; by doing a proper matching of the past frame will the present frame and in the process of matching what we will be getting is that....................the ideal situation would be that if we can generate a pixel to pixel matching but that will be highly time consuming and not very practicable simply because just by comparing a pixel’s intensity with another frame’s pixel intensity you cannot be sure that this pixel corresponds to that pixel because by fluke or by accident the two pixels can have the same intensity but only if there is a matching not only with the pixel but along with the pixel its neighborhoods also match with the pixel and its neighborhood in the other image then only we can be confident of saying that this is a proper matching. So it is not only just a pixel matching but a neighborhood matching.

Now how to specify the neighborhood?

Well, one can define a small region. But instead of specifying a region arbitrarily then again the issue will come as to how to specify the region do we take circular regions. Do we take rectangular regions; do we take any arbitrary regions? Well, the simplest of that could be a square or a rectangular region and because we are already familiar with the partitioning of an image into non-overlapping blocks, we did that in the DCT or in the transform domain coding we were subdividing the image into a set of non-overlapping blocks applying a very similar philosophy we can divide the frame into a set of non-overlapping blocks and for each block we can measure that how much of displacement it has undergone.

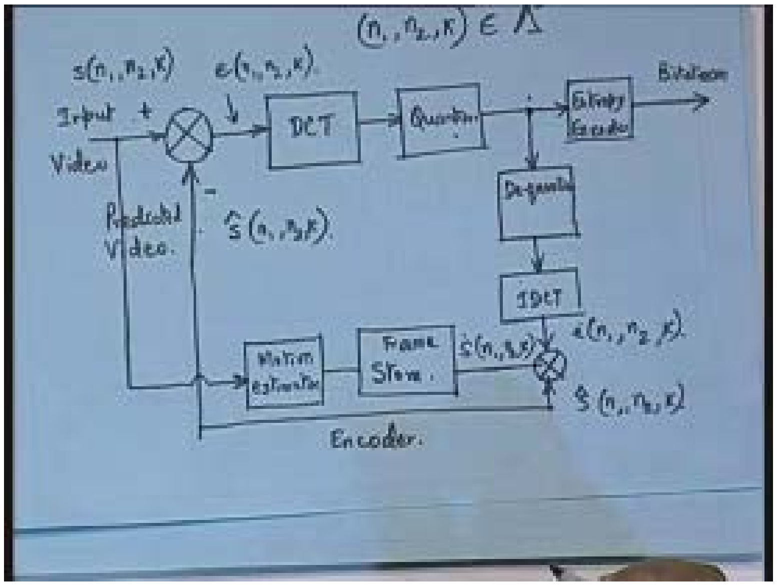
So the process of finding out this displacement it is basically a search technique and this technique is popularly known as motion estimation. This aspect was completely missing in the case of still image completions. Still image did not have anything called motion estimation but in this case, in the case of video we are going to have a motion estimation process so that the displacements undergone by every block can be measured with respect to its past frame.

(Refer Slide Time: 34:57)



What we are going to have is that this frame store output (Refer Slide Time: 35:03) has to be compared with the input video and we are going to call that block as motion estimator.

(Refer Slide Time: 35:23)

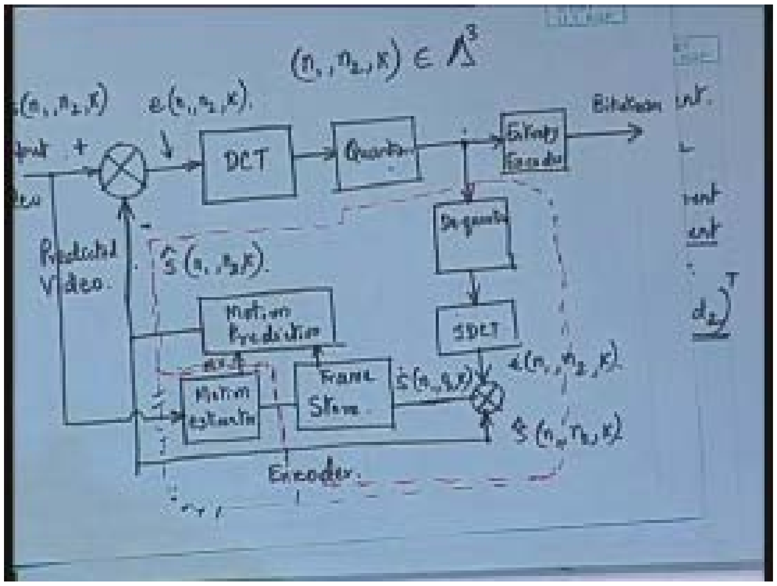


And what this motion estimator is going to generate?

The motion estimator will find out the motion vectors for every block. It will be able to generate the motion vectors for every block and this motion vector why is it useful to us? If I know that how much of motion is undergone by which block then I can fictitiously apply that motion into my stored frame and then generate the predicted frame. If I know that this is expected to move this much so I can generate a fictitious displacement and I can predict a frame. So what we do is that there will be a motion prediction motion prediction block and the motion prediction block will work on the stored frame by applying the motion vectors; I call it as MV and by applying the frame stored on to the motion prediction we can generate the predicted video. So this is how we complete the path.

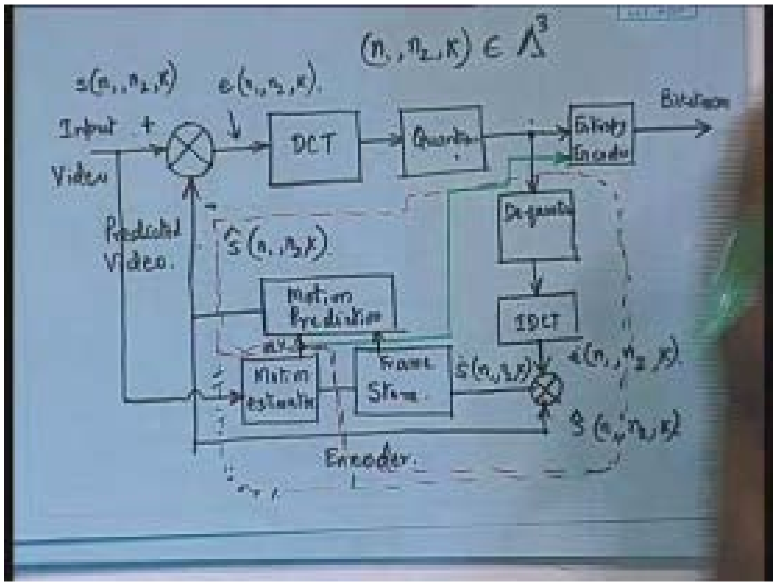
Now this is in the encoder part and if you look at this no not this sorry take out this part let me draw it in a different color. The path that I have shown within this red dotted border can also be used in the decoder; am I right?

(Refer Slide Time: 37:34)

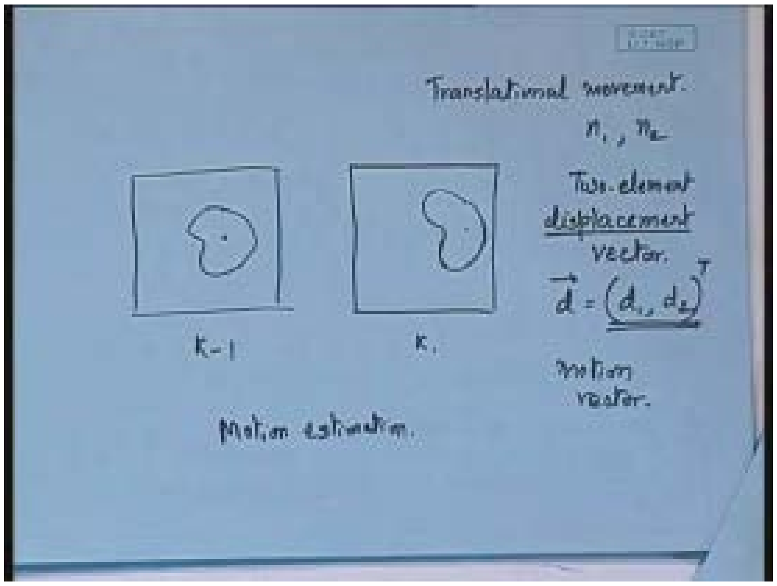


There is no difficulty; only this motion estimation I cannot do because motion estimation requires input video and I do not have input video at the decoder. For motion prediction what do we require; we require the use of the stored frame and motion vector. That means to say that the decoder must get the information about this motion vector. So this motion vector also has to be sent into the bit-stream. So we must derive a tapping from here (Refer Slide Time: 38:27) just to avoid clumsiness in the drawing I use a different color so this is the motion vector which will go into this place where the motion vector also will be having some similarities between themselves that means to say that motion vector of one block to the motion vector of the next block or in the neighboring blocks are going to have some kind of similarity so you can exploit that in the entropy encoding process so the motion vectors can go through the entropy encoder scheme and go into the bit-stream. So this bit-stream what you are containing what we are sending is not only the encoded error bit-stream but also the encoded motion vector bit-stream.

(Refer Slide Time: 00:39:17)

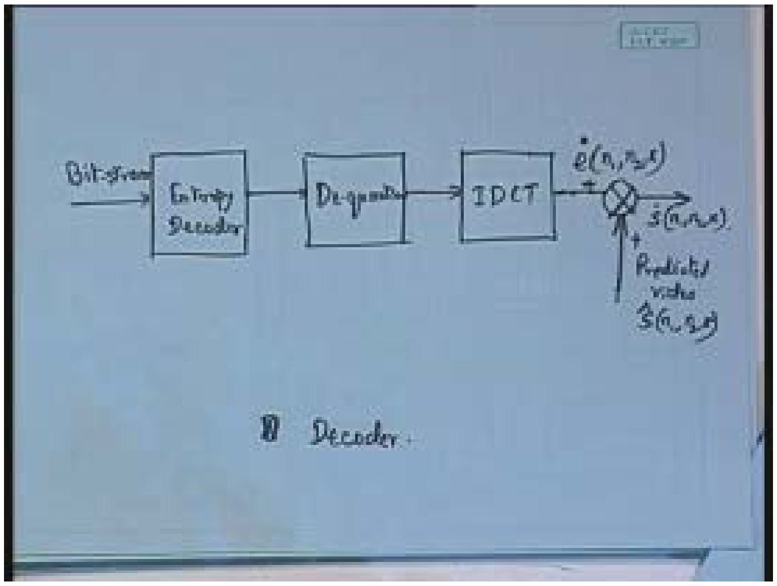


(Refer Slide Time: 00:39:22)



Therefore at the decoder block diagram this bit-stream can have the two components: one is the motion vector and the other is the signal.

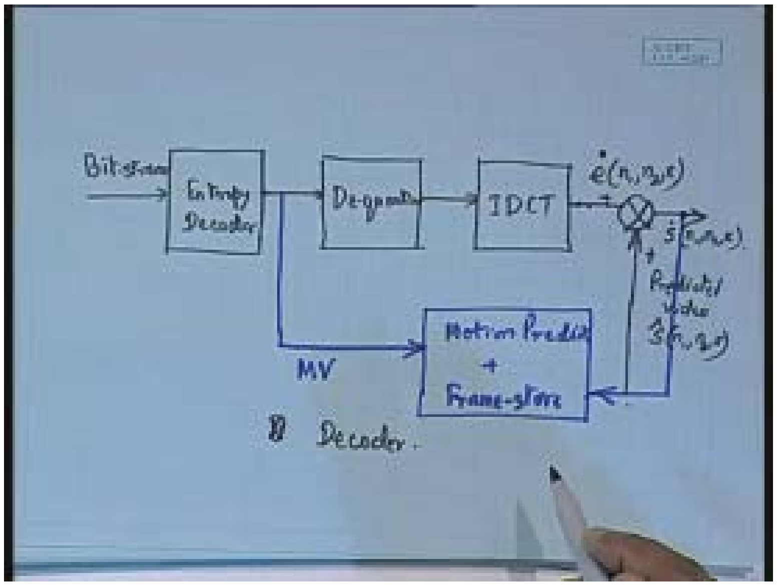
(Refer Slide Time: 39:27)



Now, as far as the signal is concerned so after the entropy decoding we make a bifurcation. The signal part goes to the de-quantizer and IDCT to derive this e dot (n 1, n 2, k) and the motion vector that can be used in this motion vector motion prediction and frame store. So I can in fact draw it as a single block so motion prediction plus frame store.

Now what does frame store require?
Frame store requires s dot (n 1, n 2, k). In the encoder it required s dot (n 1, n 2, k); decoder also we have got s dot (n 1, n 2, k) is available from here so s dot (n 1, n 2, k) could be fed to the frame store block, the motion vector is coming over here so the motion prediction plus frame store could be done from here and this is generating the predicted video s cap (n 1, n 2, k) just like the way it is done even here also (Refer Slide Time: 40:55). All that we have done is that this frame store and motion prediction we have combined into one single block. But the output of the motion prediction block is the predicted video s cap (n 1, n 2, k). So even here also it will be s cap (n 1, n 2, k).

(Refer Slide Time: 41:14)

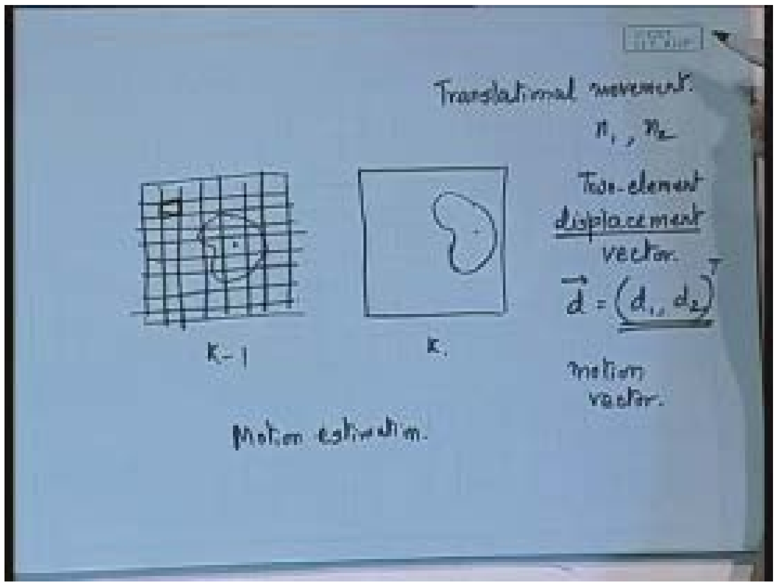


This is the complete decoder. This is the basic philosophy behind the hybrid video coding. Anybody having any doubts at this stage, before we proceed further into this? Yes [Conversation between Student and Professor – Not audible ((00:41:39 min))] Well, the question that you are asking basically goes back to the issue of transmission of both audio and video in a synchronized form, so that is a different problem altogether which is addressed as a part of the multimedia communication so I am not addressing that issue in this course but it is a very intricate issue that yes, the audio and video bit-streams which are independently generated they have to be properly synchronized and then multiplexed into the bit-stream and there are techniques there are standards for that so one has to follow those to do this.

The idea that I am pursuing here is only pertaining to the video coding. Now there are a few issues that need discussion at this stage. Now what are the new elements that we learnt from this block diagram? One major aspect that we have learnt is that a very efficient motion estimation technique has to be carried out at the encoder. Why efficient; because you see, as we had seen just by reference to an example like this (Refer Slide Time: 43:28) that we have to determine the motion vectors for every individual blocks.

Now if I subdivide this image into non-overlapping blocks like this let us say of 8 by 8 size; supposing I subdivide the image into 8 by 8 non-over lapping blocks like this and then I estimate the motion vector for each individual block just try to see the computational intricacies which are involved in that.

(Refer Slide Time: 44:05)



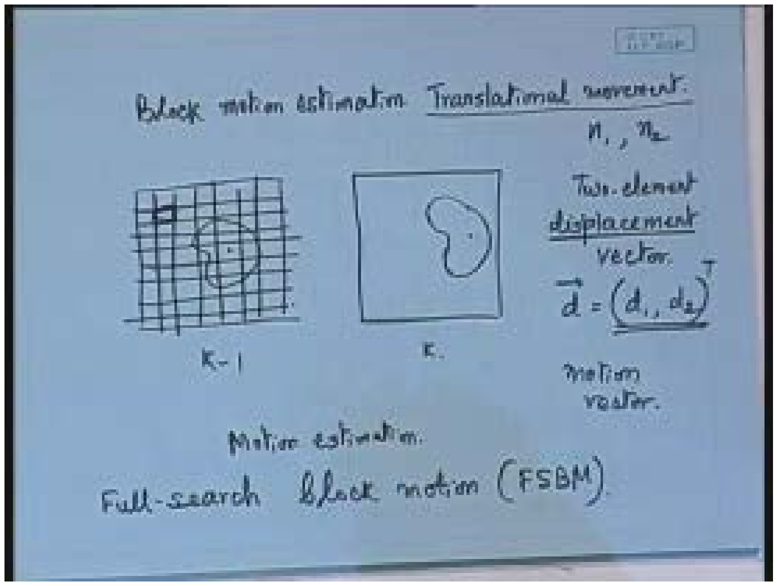
To find out the motion vector we have to do a matching and we do not know a priori that whether this particular block of pixels have move to the left or to the right or to the top or to the bottom; it can move in any of these directions I mean, assuming a translational movement; it has got two degrees of freedom, it can move in positive x negative x positive y negative y. That means to say that one has to have a search and the search is complicated in the sense that whenever we are matching one block with a block that is present in the past frame, the past frame is actually used as a reference.

In this case, in the block diagram what we have essentially done is that the image that we obtained from the frame store that is the past reference frames whereas the input video is being called as the candidate frame and this is the past frame. Candidate frame is one which we are going to encode (Refer Slide Time: 45:22) and past frame is what we are using as a reference so the two terms that we use is the candidate and the reference.

Now, what we have to do while searching is that we have to search within a range and then when we have to search, for every position we have to find out that what is the similarity in this block of pixels with that in the reference frame; candidate block to be compared with a reference block now and if the block size is 8 by 8 then we have to compare the values for 64 pixels 8 by 8 that is 64 pixels and that we have to do for only one search position but there will be several search positions.

If I specify a search rate that okay, any unlimited search range one need not have to use because as I have told that the movement between the consecutive frames are going to be quite restricted that is why we need not have to search over a very long search range or very large search range, we can restrict the search range. But even within the restricted search range also we have to do lot of searches. So there will be good computational issues that will be involved and there are some techniques in order to have very efficient search technique computation; although the optimal search technique is always the one which is carried out at exacting search or what is known as the full search block motion. This way of estimating the motion by subdividing the pixel into blocks like this it is called as the block matching technique or block motion estimation block motion estimation technique whereby we are subdividing the image into small blocks and estimating its motion and whenever we are searching over the entire search range and estimating the motion we call it as full search block motion or in short form it is called as FSBM.

(Refer Slide Time: 48:07)



Now FSBM guarantees that you will be able to find out the optimal position at which the search is the best. Now how to quantify that search or rather how to measure the quality of that search that I will cover in the next lecture and then we will also talk about some efficient techniques for that which had been used in some of the popular standards and we will also talk about some of the popular video compression standards, thank you.