**Digital Voice and Picture Communication**

**Prof. S. Sengupta**

**Department of Electronics and Communication Engineering**

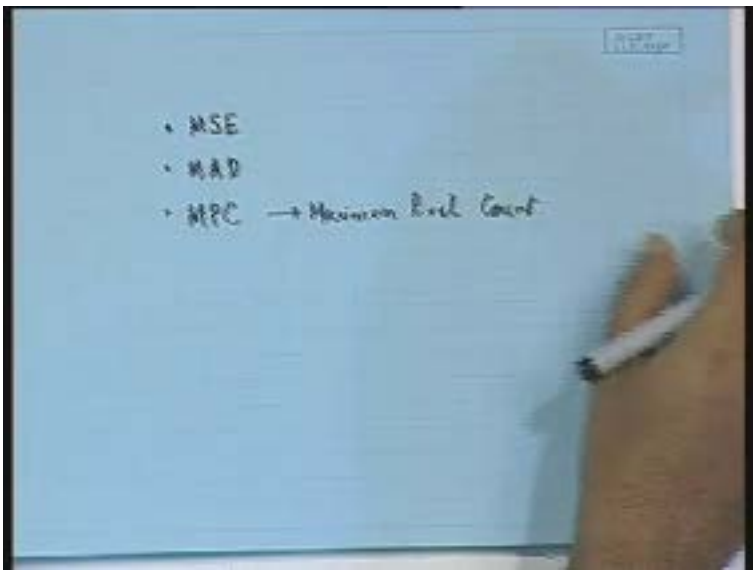
**Indian Institute of Technology, Kharagpur**

**Lecture - 25**

**Fast Motion Estimation Techniques**

So we were discussing about the different aspects of motion estimation and towards the end of the class we were discussing about some of the matching measures in order to determine the motion vector. So, two of the measures which we had discussed they are the mean square error or finding out the minimum mean square error and the other criteria that we talked of was the minimum of mean absolute difference so MSE criterion and MAD criterion and we were also mentioned that there is a third criterion which one uses and that is the MPC which stands for the maximum pixel count.

(Refer Slide Time: 1:51)

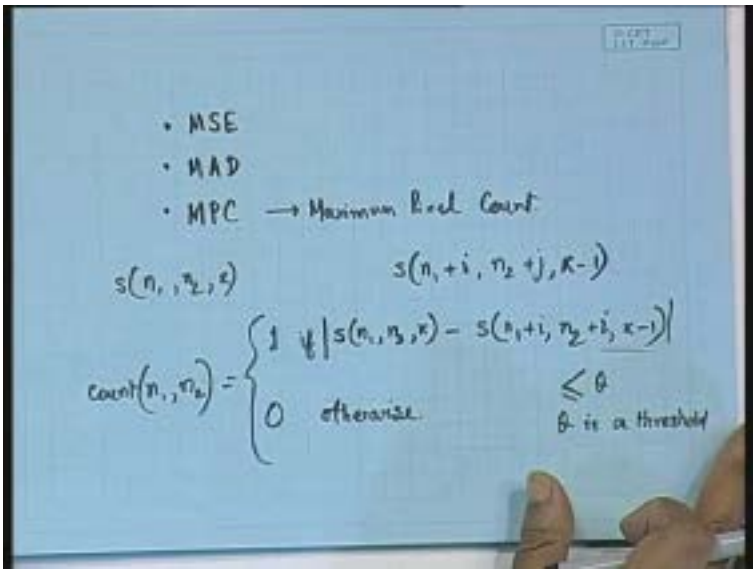



Now this is actually a matching measure and what we mean by maximum pixel count is that, if we take the image function as s (n 1, n 2, k) and in the reference frame we search in the space

where the displacement is assumed to be (i, j) that is to say that the displacement vector being i and j we will be searching there at (n 1, plus i, n 2, plus j and k minus 1) because k is the current frame and k minus 1 happens to be the reference frame and then in that case we can decide that if this quantity that is s (n 1, n 2, k) minus s (n 1, plus i, n 2, plus i, k minus 1) if the mod of this quantity is less than or equal to some threshold theta some pre-assigned threshold so theta is a threshold. So if it is less than a particular threshold that means to say that the pixel at (n 1, n 2) for the frame k has matched well with the pixel at the position (n 1, plus i, n 2, plus j at the reference frame k minus 1).

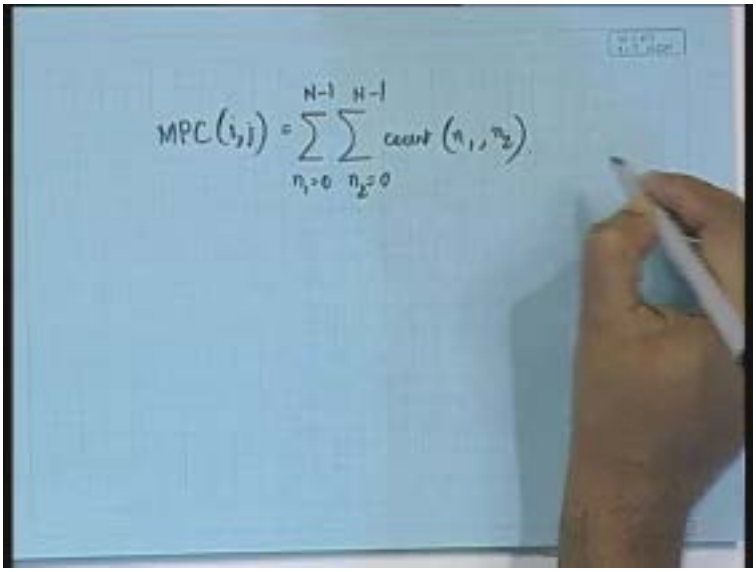
In such case we can say that on a pixel to pixel basis as if to say that we have found a match. So what we do is that we keep a count so we keep a count at the position (n 1, n 2). So count (n 1, n 2) we just define a binary quantity and that binary quantity assumes a value of 1 if this magnitude difference between these two quantities that is for the current frame k and the reference frame k minus 1 if this is less than or equal to theta in that case we put the count value to be 1 otherwise we put the count value to be equal to 0. So, count is equal to 0 obviously means that there is no matching. The pixel value here is far deviated with respect to the corresponding pixel value at this position in the past frame.

(Refer Slide Time: 4:40)



Therefore, what we will typically look for is that, if we assume a block again and assume that (n 1, n 2) are the coordinates belonging to the block B..... so B is the block of pixels B is the block of pixels and if we take (n 1, n 2) belonging to B in that case we can define a matching criterion based on this; that is to say we say maximum pixel count or we call it as the MPC measure corresponding to the displacement i and j and that we define as the summation or in this case double summation because we have to sum up over the n 1 where n 1 goes from 0 to capital N minus 1 and n 2 goes from 0 to capital N minus 1 the quantity count (n 1, n 2).

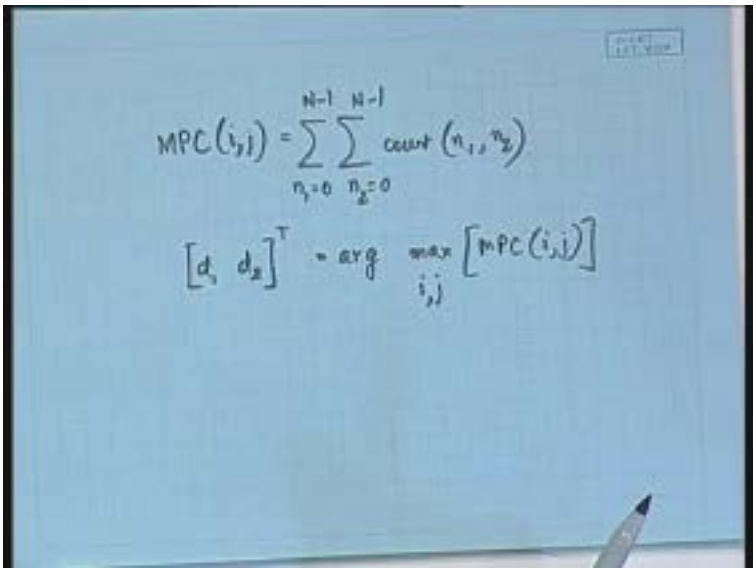
(Refer Slide Time: 5:48)




this means to say And here this measure we just find out for all values of this n 1 and n 2 where n 1 and n 2 belongs to the block (Refer Slide Time: 6:02) that means to say that if we are taking an 8 by 8 block, in that case, over all the 64 pixels we are going to adopt this count. So what we should look for; in order to get the motion vector what should we look for? Earlier in the case of MAD or MAC we were looking for the minimum measure. Now here what measure are we going to take? It is maximum measure. So in this case we should go in for the argument of the maximum value of MPC for the displacement (i, j) so just search over (i, j) and whichever (i, j) combination that means to that whichever vector motion vector gives you the maximum value of the MPC that (i, j) vector you take as the displacement vector or the motion vector. So we can

say that the motion vector d 1 d 2 (.....7:18)..... the d 1 d 2 transpose that will be given by the argument max of this quantity. Otherwise this is a very simple measure.

(Refer Slide Time: 7:28)




But no matter whether

One advantage that one can claim about the MPC is that this count is a binary quantity so it is just a hard thresholding and it is quite sensitive to this threshold. So the question that needs to be addressed is that how to pick up this threshold value because if the threshold value is not correctly picked up then the matching measure could be then the matching measure should not be trusted; otherwise it is a very simple measure, just adding up; all that we need is an adder and we do not require any multiplication. Of course we did not require any multiplication for the MAD measure as well even for the MPC.

Now what we are going to consider is that, no matter whether we consider the MAD or the MPC or the MSE measure, one thing is very certain that corresponding to every search position our computational complexity is of the order of n square where n by n happens to be the block size so the n square happens to be the order of computational complexity and if we have a search window defined as the plus minus W; we had seen yesterday that we had 2 W plus 1 by 2 W plus 1 multiplied by n square and that is a very big number even if you choose W to be a reasonably

small quantity of W equal to 7 and you choose n to be the practical as what we take for all practical applications that is we take n is equal to 8the number is quite high.

So naturally when we talk of estimating the motion in real time, definitely the full search block motion is going to have lot of computational complexity. We have only two options before us: one option is that we go in for a specially designed hardware that we use some we design some hardware unit which will compute the motion vector and there is enough of research that has gone into this and still some further researches are going on and especially in today's VLSI technology there are several efficient architectures which have been developed in the APGA or in the AZ and literature have reported that many commercial products are also available but other than using the specialized hardware the other solution is to go in for a fast search methodology or a quick search methodology. So, that will be what we are going to discuss in this lecture.

In this lecture we are going to talk about the fast motion estimation technique. And why fast techniques are required; that I have already explained. In fact, you yourself should have realized this by looking at the computational complexity that is involved in the usual full search block motion applications.

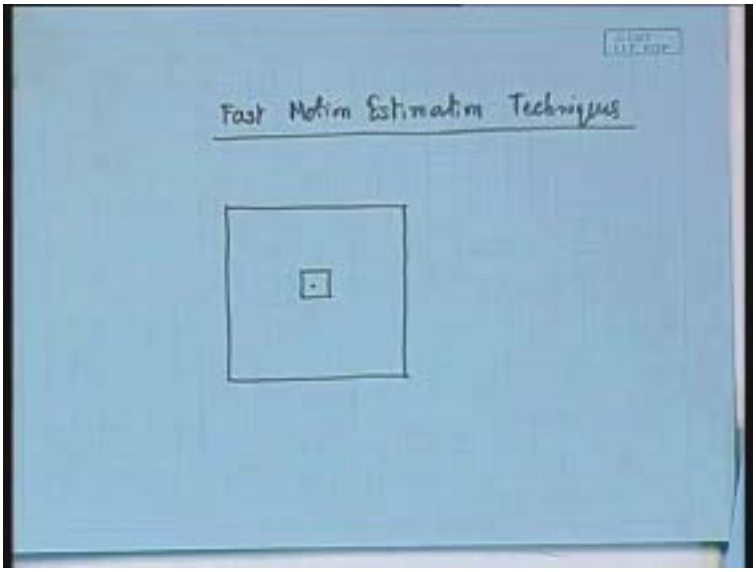
In the case of fast motion estimation techniques, what we have to do is to reduce the number of computations that we are going to do. Now, one point is very clear that in the case of the full search block motion, although the computational complexity is very high but we are guaranteed to get the minimum solution. Unless we have gone very wrong in choosing the maximum window size the search window size, if we say that W is 7 and we search within the range of plus minus 7 but if the actual motion happens to be beyond plus minus 7 pixels then we are wrong; but otherwise the full search block motion is guaranteed to give you the optimal solution.

But in the case of fast techniques let me question you that fast techniques will be computationally efficient but it is not guaranteed to give you the minimum solution. You may have to feel satisfied with a kind of sub-optimal solution because it is not going to check or it is not going to compute the matching measure in all positions within a window. In this case in the

case any matching measure here (Refer Slide Time: 13:06) which is the value of the (i, j) that we use for searching; the entire search widow. If it is the full search block motion we will search everywhere within the search window. But in the case of fast search we do not do that. We only search at some specific points and those specific points are to be very judiciously chosen.

How to choose that specific points; for that algorithms are existing and we are going to talk about some very fundamental and popular algorithms that people have suggested. Now, the motivation behind the fast motion estimation technique is that, one can go in for..... as you go away from the actual minimum position..... to illustrate that let us take some search window.

(Refer Slide Time: 14:38)

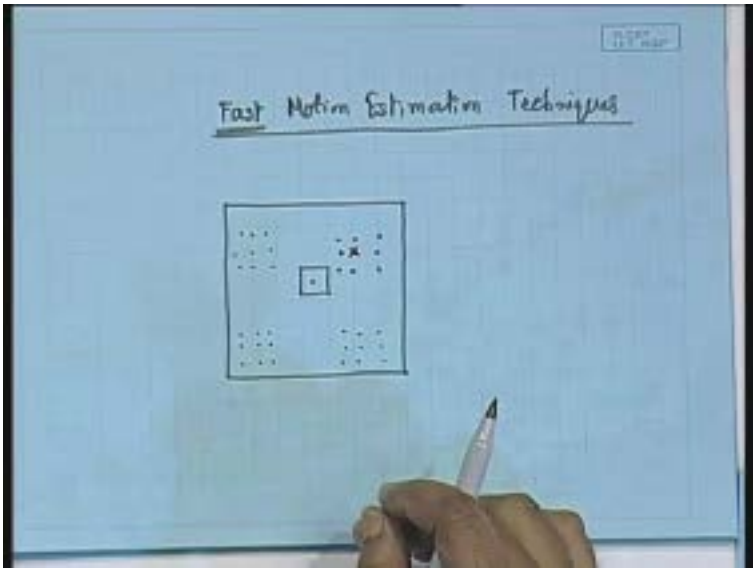



Now supposing this is a search window and this is the centre of the search window (Refer Slide Time: 14:26) and we have first placed a candidate block at the center of the search window and then we search at different positions within the search window so that the candidate block will be placed in different positions over here and then we are going to find out the matching measure no matter whether we take MSE or MAD or MPC all these kinds of criterions that we talked of. So the heuristic that is applied for the fast motion estimation technique is that, as you go away from the actual minimum..... let us say that this being the search window (Refer Slide Time:

15:06) and this being the centre of the search window there is some position let us say over here where the minimum actually happens; minimum means that this is the true search, the minimum of absolute difference would lie at this position.

Now the heuristics says that okay, in this case at this point you are going to reach the minimum. Now supposing you do not choose this point for your matching then are you going to lose everything? No.

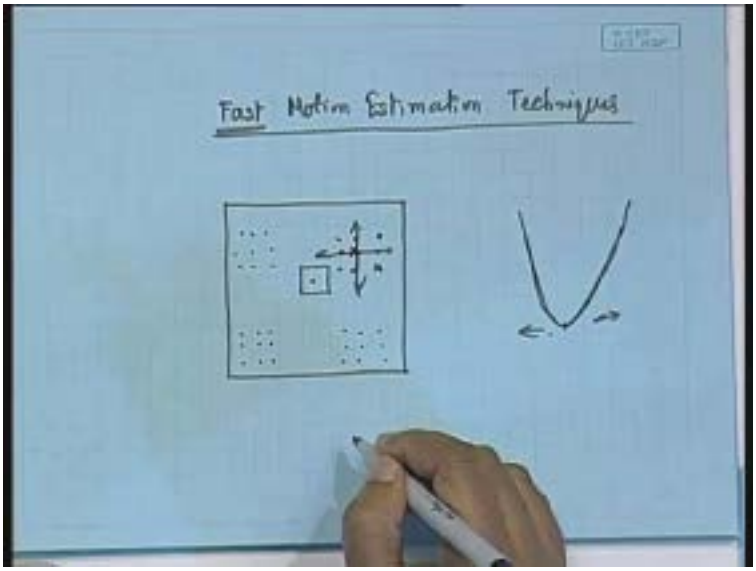
(Refer Slide Time: 16:31)




If you are searching in the close vicinity of the point of minimum, you will be still finding some lower values. If you search at this locations here you are going to find lower values as compared to may be somewhere over here at this search positions where you are going to find higher values; even here also you may find higher values of matching measures, at all other positions you are going to find higher values of matching measure. here you are going to find the minimum (Refer Slide Time: 16:16) but close to this close to the minimum you are going to find lower values and as you go away from the point of minimum, the assumption that people make is that the matching measure value if you are using the MAD or the MSE, it monotonically increases; which means to say that if this the point of minimum, then as you go away from the minimum it will monotonically increase, the measure will monotonically increase at both the

sides. That means to say that whether you go here or here (Refer Slide Time: 17:00) or up or down, left or right, up or down everywhere you are going to find an increase in the value and that is monotony so that once you are close to the point of minimum you are going to find out quite low values then the search strategy could be decided this way that okay, if you do the searching at some specific points and obtain some minimum out of those search positions, take that minimum position and then you refine your search space.

(Refer Slide Time: 17:51)




Once you find that your tentative minimum is obtained somewhere over here then you did not have to search over this entire range; you reduce your search range, perhaps you search in this position. If you then find that your minimum happens to be here (Refer Slide Time: 18:08) you reduce your search range further until the time until the point you can come to the specific point of minimum. So, at least it guarantees or at least it tries to see that, in a few iterations you will be at least close to your point of minimum or at least very close to your point of minimum. But if you are not exactly at the point of minimum you cannot call it as an optimal algorithm. But if you are close to your point of minimum you can at least call it as a sub-optimal search technique.

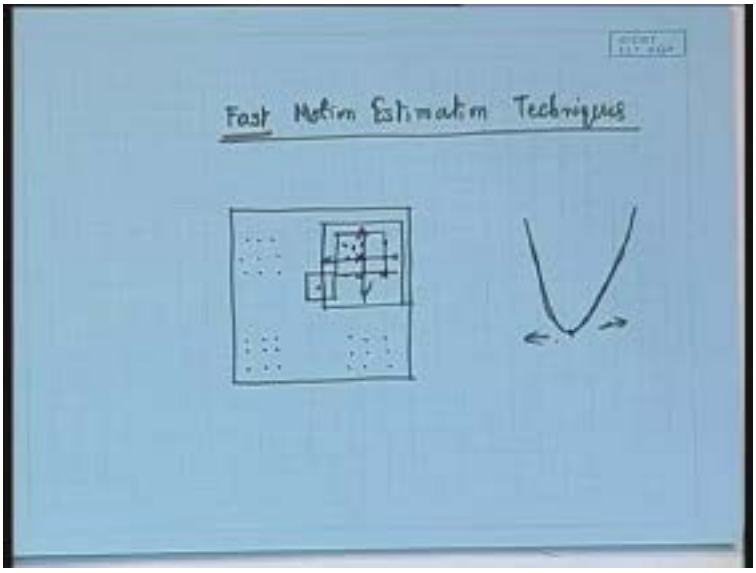
Now there is no direct mathematical proof which people can give to this. If you ask me that is there a mathematical proof that this is going to happen that it is going to be monotonically

increasing as we go away from the point of minimum, yes, the answer is no because you never know that how the image is going to be and not the image, here there is involvement of two images: the candidate image and also you have the reference image. So unless you know the nature or the intensity values of this you cannot really guarantee that this will happen. But it is seen it has been observed that it happens. And intuitively we can still feel that why it should happen because once we are at the minimum, definitely the intensity values very properly match so we are going have a minimum value of the MAD or minimum value of the MSE and if we are very close, the intensity values have not changed drastically.

Again this is expected from the smoothness criterion of the images; that the intensities in all natural image that follows a smooth function. So the only place where we can expect a deviation from this heuristical approach is where you have very sharp discontinuities in the image intensities. Yes, when you have discontinuities then this heuristic may not hold good. Anyway so with their basic assumption that for most of the images this heuristic would hold good; some efficient search techniques, suboptimal search techniques or quick search techniques can be described and one of that is what has been described by Jain and Jain.

Jain and Jain's technique is referred to as 2-D logarithmic search 2-D logarithmic search. In the case of 2-D logarithmic search what is done is that we assume a search range, a maximum search range is assumed; let us say that the search range is p so the initial search range is p and let us say that p is equal to 8, 8 means that it could be 8 on this side, 8 on this side, 8 on this side and 8 on this side.

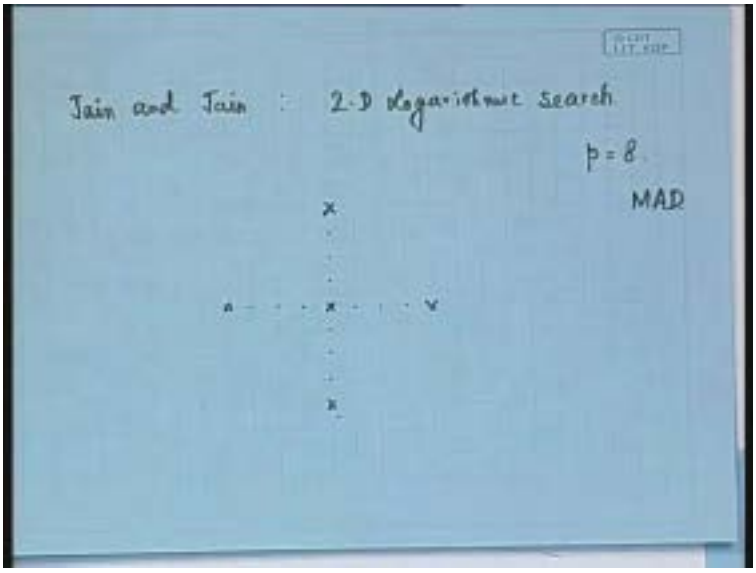
(Refer Slide Time: 20:15)




Now, in order to search for the motion vector, make a quick search of the motion vector supposing this is the center of the search window center of the search window means where our candidate block is actually located, the coordinate position at which the candidate block is located that is the center of the search position and if it is a p is equal to 8 in that case at the positions p by 2 away from the center of the search; in this case p by 2 is going to be 4 so we go left by four positions 1 2 3 4 so at this position, again 1 2 3 4 at this position (Refer Slide Time: 22:39) this one and at this position so there are how many positions I marked? I marked 1 2 3 4 5 so five because the center is also considered so we search only at these five positions and not anywhere else in the first step of search.

So search window p is equal to 8 obviously means that we are going to have 2 into 8 plus 1 2 into 8 plus 1 that means to say that 17 by 17 as our search window size in the full search block motion case 289; 17 into 17 means 289. But instead of 289 we are searching only in five positions.

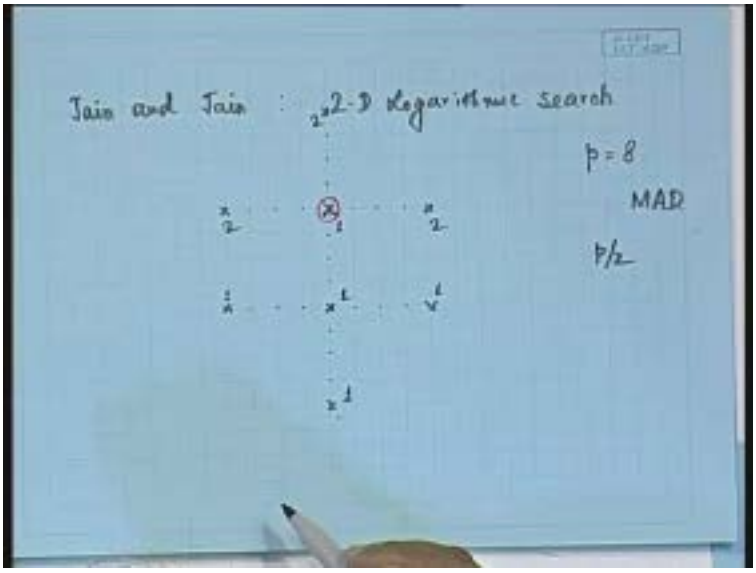
(Refer Slide Time: 23:45)




Now let us search in these five positions; let us assume that we have MAD as the matching measure. So out of these five positions you find out that which one gives the minimum. Supposing the minimum is obtained at this position I encircle this (Refer Slide Time: 24:06). This is the first minimum position that has been obtained.

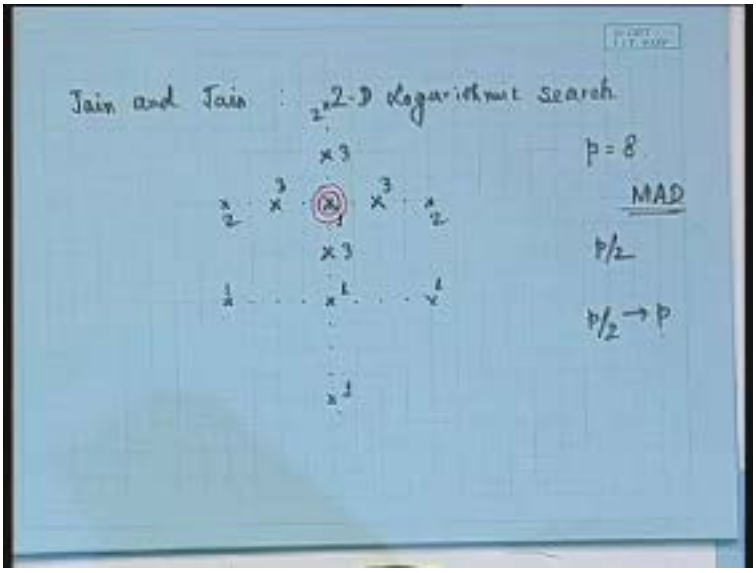
Now what happens is that we carry out our search further by taking this newly obtained minimum as the reference. Again we consider the step size of p by 2. So just like the way we did it earlier we take it as 2 3 4 so this position; 1 2 3 4 so this position,1 2 3 4 so this position (Refer Slide Time: 24:48) so we are going to again search in 1 2 3 4 5 position these five positions; of course out of the five positions here we do not require any fresh computation because earlier itself in the first place itself i marked them as 1, so in the first iteration itself I have already picked up this one and this one and in the second iteration we have to look for these three new (Refer Slide Time: 25:19) but anyway in order to compare that which out of these five is a minimum we compare between these five values and we have to obtain a minimum.

(Refer Slide Time: 25:32)




Now, if the minimum happens to be the second iteration minimum happens to be this one in that case we will continue our search further. But if out of these five positions the minimum happens to be again at the center; look at this thing that if the second minimum if the second iteration minimum (Refer Slide Time: 26:01) again happens to be the center of this newly defined search window there what we do is that we reduce the step size further by a factor of 2. So we make p as p by 2, so p by 2 then becomes new p that means to say that our search step size instead of being 4 over here we make it as 2 which means to say that if this is the second minimum position if this is the second minimum position then for the third search we will be using this one, this one, this one and this one so we mark them with a number 3 written on its side. So all these four positions written with number 3 on the side these are the pixels which are to be these are the positions at which we need to compute the MAD matching measure. So these four positions plus the center one value, out of these five we decide whichever is the minimum.

(Refer Slide Time: 27:36)



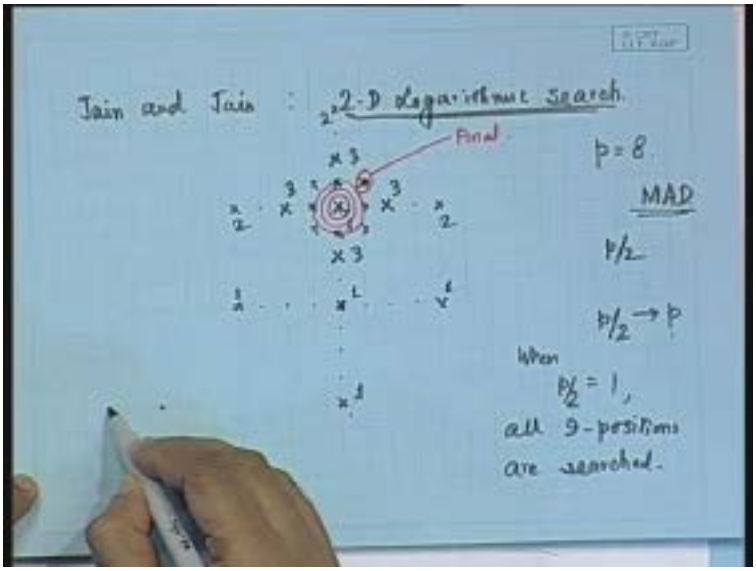

If the minimum happens to be something other than the center we continue the search taking the same old value of p. But if the minimum happens to be the center, every time the minimum happens to be the center we reduce p by a factor of 2 which means to say that if for the third iteration also this is the position of the minimum in that case my step size gets halved further which means to say that, in that case my step size becomes equal to..... p by 2 becomes equal to 1 then in the fourth iteration I must search this one, this one, this one, this one.

But according to this 2-D logarithmic search algorithm when p by 2 becomes equal to 1 when p by 2 becomes equal to 1 that means to say that the search step size reduces to 1, in that case all nine positions are searched, all nine positions all nine neighborhood positions are searched. What I mean by that is that if p by 2 happens to be one as is happening in this case there it is considered..... p by 2 is equal to 1 means there cannot be any further iteration of search, this is the last iteration of search because you cannot make p as a fraction because the step size has to be an integer so you are searching within these nine positions in the fourth iteration and out of these nine whichever gives you the minimum that is what you declare as the final minimum. If the final minimum happens to be this one then this is the final. So in that case the motion vector will be given as the coordinate position of this with respect to this or the displaced

coordinate position between this one and this one that will be the final motion vector which would be decided accordingly.

Now why it is 2-D? Simple; the search space is two dimensional and why is it logarithmic because the step size is logarithmically reduced. At every step, at every iteration we are reducing the step size by a factor of 2 and that is why it is known as logarithmic search.

(Refer Slide Time: 30:26)

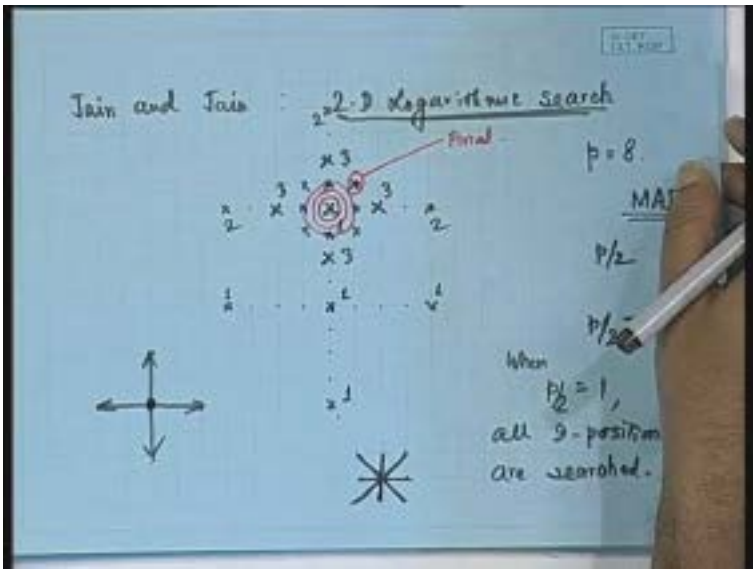



Now look at the beauty of the logarithmic search technique. If this is the center of the position we search at the end of a plus (Refer Slide Time: 30:42) if this is a center of the search then we search at the end of a plus sign where the length of the plus sign will be decided by what value of p we have. Only at the last step we are searching all the nine all the nine neighbors, so there we are not only searching the plus but also searching the cross neighbors; plus neighbors as well as the cross neighbors all are searched only in the last step.

Now, there is a variant of this algorithm. In fact there are several variants of this fast search techniques which have been proposed in the past and of that I can tell you about one other technique which is known as the cross search. Now, as the name suggests, cross search means that instead of searching around the plus you search around the cross which means to say that if

this is the center position you have some value of p and then you go by steps p by 2 on either side in the cross direction that means to say here one step, two step, three step, four step so let us say that p by 2 is equal to 4 so you search over here 1 2 3 4, you search over here (Refer Slide Time: 32:25) 1 2 3 4 here and you search here so here also you have got five different positions at which you are going to search.

(Refer Slide Time: 31:28)

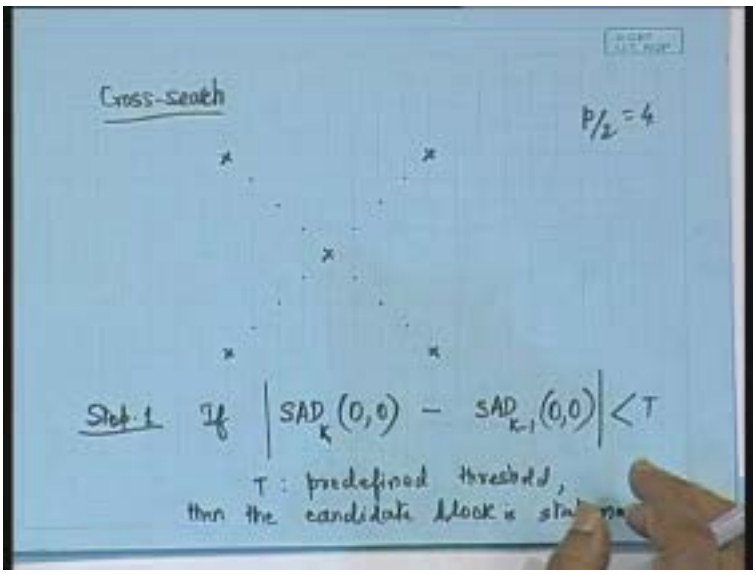



And a very similar algorithm is employed. Now the only now the there are two major modifications which have been done in the cross search algorithm with respect to Jain and Jain. Of course, changing from the ends of a plus to end of a cross is not a very big change for 7 because depends on how one defines the neighbors. But one important aspect which have been added in the cross search algorithm is just an application of a thresholding to see that, I mean, before beginning the search at all you find that at the (0, 0) position; (0, 0) position means that at the candidate block position if you compare the sum of absolute difference measures of the candidate block position with that of the candidate block position with that of the reference block at the (0, 0) position at the same at the candidate block position that means to say that there is no movement; so if it happens that the difference in the sum of absolute difference values at this position in the frame k and the frame k plus 1 they happen to be less than a predefined threshold,

in that case you need not have to search any further, you can declare that this block is a non-moving block.

Why it is being done is in the case of background or in the case of many stationary objects it is quite often observed that one does not have any significant motion in those regions. So, if there is no significant motion then why to use a technique where after applying three iterations or after applying four iterations you finally conclude that no, the best position is (0, 0) that means to say that the biggest motion vector is (0, 0) so there is no change no displacement. So why to declare that after doing some computation; if that is the case why not you declare it early. So this is exactly what is being done in the cross search. So what we do as the step 1 of the algorithm is that, in the step 1 we calculate the sum of absolute difference measure for the frame k at the position (0, 0) and then we also find out that what was the sum of absolute difference value by taking k minus 1 as the candidate at the same (0, 0) position and if the difference in mod of these two values happen to be less than our threshold so T is the predefined threshold so T is the predefined threshold and if this value happens to be less than T then the block is non-moving then the candidate block is a stationary block.

(Refer Slide Time: 36:44)



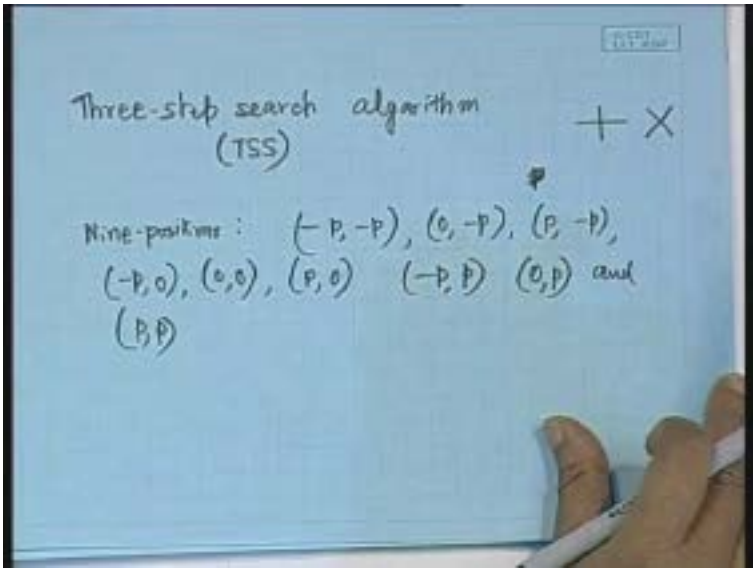
If it is a stationary block you can terminate search at the step 1 itself, you need not have to go further. But if you find that it is not the case that means to say that it is a moving block only then you go in for the motion estimation. And in the case of motion estimation the approach is taken very similarly that means to say that you search at the ends of the cross; instead of searching at the end of a plus you search at the end of the cross and then you find out that if it happens to be the center then you reduce the search step size by a factor of 2 and then this algorithm..... I mean, at the last stage when p becomes equal to 1 it does not go in for a nine point search but instead it takes up that you either going for the end of a cross or you going for the end of a plus depending upon whichever is the minimum.

Now I am not going into the details of that algorithm because very simply it is not possible to exactly cover all the fast search techniques there are many literature references which are available for this. The idea is only to introduce you to different search efficient search strategy algorithms.

Now there is yet another very popular fast search approach which people have worked out and that is the three step search algorithm three step search algorithm. The three step search is very popular and in a short form it is referred to as the TSS algorithm. Now, in the case of TSS one can say that it is a combination of the cross search and the 2-D logarithmic search, 2-D logarithmic search at the end of a plus and cross search is the end of a cross and in this case in the in the case of three step search algorithm the searching is actually done in all the nine positions. So there what you do is that you define some initial value of p and then at the position so then you search at nine positions. And what are those nine positions? Those nine positions would happen to be minus p minus p assuming that the total search range in this case is going to be 2 p plus 1.

The nine positions are minus (p, minus p) (0, minus p) (p, minus p) (minus p, 0) (0, 0) (p, 0) (minus p, p) (0, p) and (p, p) these are the nine coordinate positions 3 4 5 6 7 8 9.

(Refer Slide Time: 40:16)

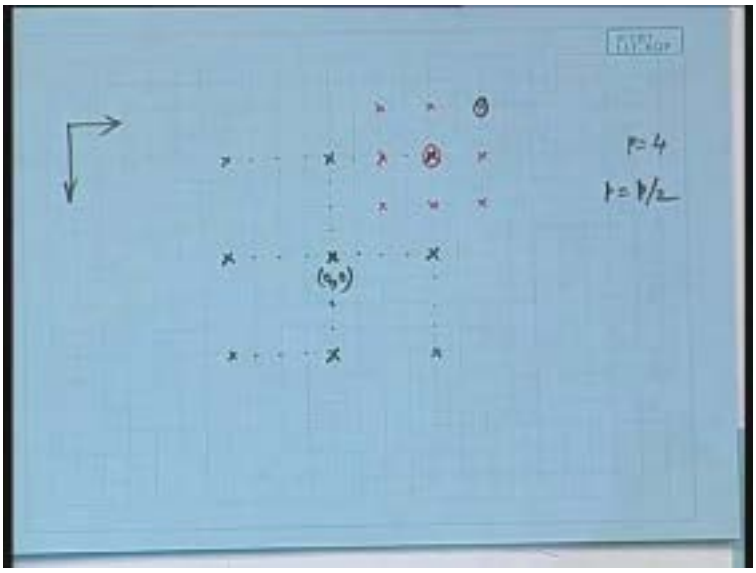



Now just pictorially it will be better. If this is your (0, 0) position and let us say p is equal to 4; 1 2 3 4 so this is (minus 4, 0) or (minus p, 0) so this is one search position 1 2 3 4 so this is (p 0); we have all these things so we have shown (minus p, 0) we have shown (p, 0) then what we have to show, the end of plus if you say so then it is (0, minus p) and this is 2 3 4 so this is (0, plus p) so if this is the plus direction and if this is the plus direction that means to say that this way if we follow the convention that means to say that going this way makes it positive and going this way makes it positive (Refer Slide Time: 41:17) so in that case this is (p, 0) and this is minus no, this is (0, p) is this one and this is (0, minus p). So (0, p) and (0, minus p) they are also covered over here and then we have 1 2 3 4 positions from here, 1 2 3 4 positions from here, 1 2 3 4 over here, 1 2 3 4 over here so totally we have including the center we have got 3 here, 3 here, 3 here so 6 9 nine search positions then we search at all the nine positions and determine that which is the minimum value and at the minimum value position let us say that this is the minimum value in that case you need not have to wait for the reduction of the search space but just you find one minimum and then you reduce p by factor of 2.

Therefore, as soon as you find any minimum you make the new p as p by 2 and there what you do is that now your search space now becomes with 2. So in the second iteration you search, I

use a different pen here, here, here, here, here, here, here and here again nine positions and of course the center is also included.

(Refer Slide Time: 42:55)




Now let us say that we have found the minimum at this position. Now, we once again make p divided by 2 which means to say that p is already 2 so 2 divided by 2 makes it 1. So with this as the minimum position we will be searching 1 2 3 4 5 6 7 8 9 at these nine positions and out these nine positions whichever gives us the minimum value if this happens to be the minimum value, this will be our final position.

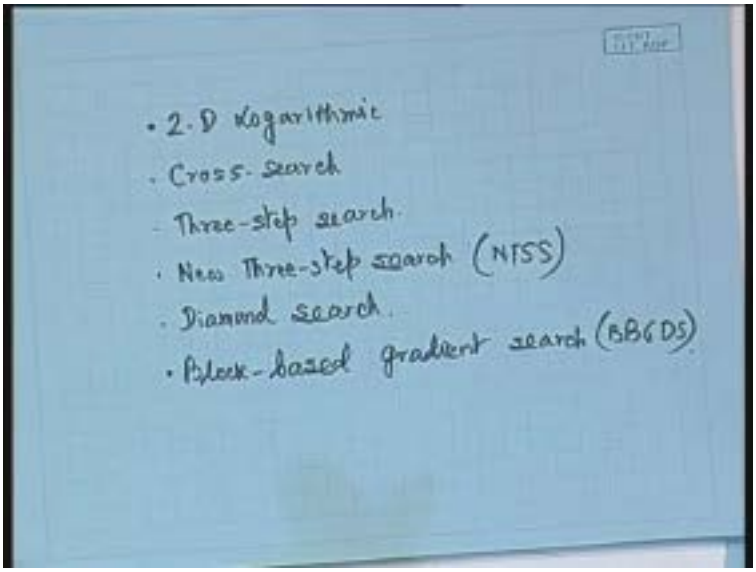
Now, in this case in the example I took p is equal to 4 and that leads to a search space of 2 p plus 1 that is to say 9 by 9, a reasonably okay search position and in this case the number of steps that you perform is clearly 3. Of course the people who originally proposed this algorithm they had named it as a three step search position but of course if you think in a very general way you need not have to call it as a three step search, you can call it as yet another form of a logarithmic search because if instead of p is equal to 4 you assume p is equal to 16 let us say in that case it becomes a five step search because p is equal to 16 means you will require five steps to bring down to p is equal to 1 so your algorithm stops only when p is equal to one and you search all the nine positions that will be the last part of your search. So some of the techniques so just to

summarize some of the fast search techniques that we have reviewed today are the 2-D logarithmic search, we have talked of the cross search, we have talked about the three step search and if you are keen to explore the literature you will be finding that there are many different variants of these techniques.

Some of the other techniques that I can just briefly make a mention is an algorithm called new three step search which is an efficient form of three step search called as NTSS and then the diamond search algorithm and then gradient decent search block-based gradient decent search; these are some of the techniques that one would find in the literature.

In fact the reason wise so much of research has gone in the past several years is that motion estimation happens to be the most time consuming block in the video codec architecture. So the critical path actually lies there and that is why the research community is so keen to develop efficient search strategy in order to reduce the search time. Again the algorithm should be such that it reduces the hardware complexity, the algorithm should reduce the computational burden, so with all these tradeoffs in mind many efficient search techniques were developed and all these search techniques all these fast motion estimation algorithms and everything have gone into the various forms of the video codecs that have come into the market so far and some of which are still coming.

(Refer Slide Time: 46:26)



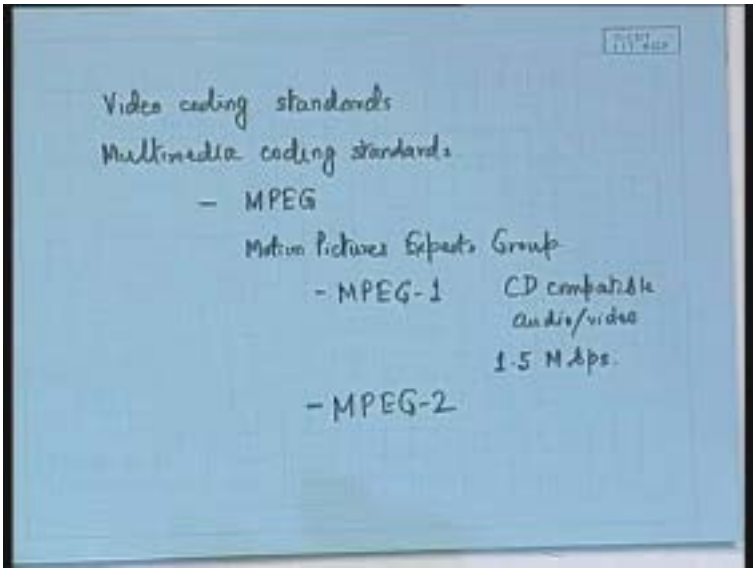

Now, before I conclude the lectures on the video coding I should also make a brief mention about the different video coding standards that have been designed till date. So just like the way whenever we are transmitting the still images we have the standard JPEG and the improved version of standard is the JPEG 2000 as we already mentioned.

For the video coding actually there are several standards which have been proposed till date. In fact when we talk about video there are two kinds of perception which people have: one, community who feels that video would mean that it is only the continuous sequence of images. But there is also another kind of perception where they say that not only images, not only the sequence of images but also the accompanying audio part whenever you are filming something. Even when you are using your handy cam, your own camera, your own movie camera when you are handling, there, you are interested in both audio and the video parts together. So in that case it is a combination of more than one media stream; one media stream is the video and the other media stream is the audio and this leads to multimedia streams; a combination of multimedia bit-streams. So, video coding standards also result in the development of the multimedia coding standards. Although the difference is quite narrow down but there are two major communities in the world who had looked into this video coding standards and multimedia coding standards.

As far as the multimedia coding standards are concerned the research community basically looked into the standards which are referred to as the MPEG standards. The full form of MPEG as most of you will be knowing is the Motion Pictures Experts Group. So MPEG was the body who had formulated the multimedia coding standards. So when they said multimedia coding standards naturally it is not the video alone but video along with audio that is what they have to cater for their standard and even the aspects like the synchronization between the audio and the video and all these things.

Now, the first MPEG committee was formed way back in the year 1988 and over the past 18 years lot many MPEG standards have been proposed. And in the MPEG, actually the first version, the first of these MPEG standards was the MPEG-1 that was the very first standard and MPEG-1 basically addressed the question of storing the videos into the CD so it looked into CD compatible audio and video storage CD compatible audio and video storage and there the target bit rate that was specified was 1.5 megabits per second. So this was the first of the multimedia standards and then the second standard that came into the market that was upgraded the second upgrading was done as the MPEG-2 so that was the next MPEG standard. But MPEG-2 charter was quite different; MPEG-2 actually addressed not only the lower end of application but also it addressed the very high end of applications like the HDTV the High Definition Television standards.

(Refer Slide Time: 52:34)



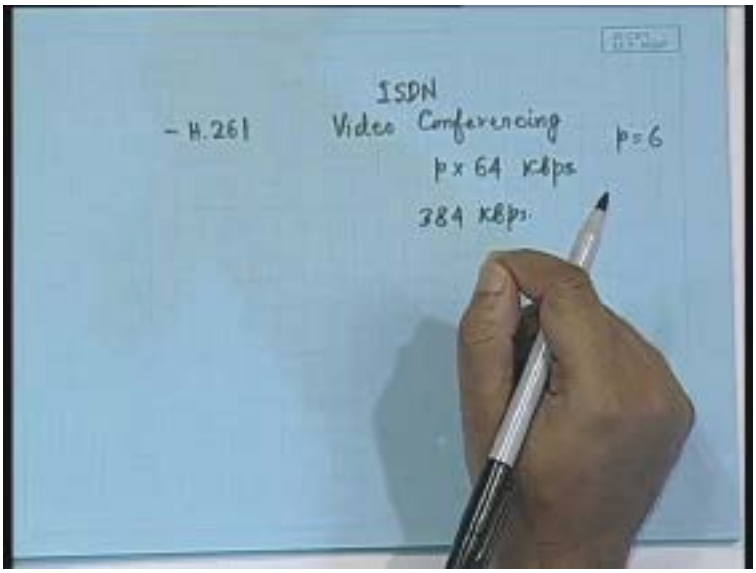

HDTV requires a much wider bandwidth that is why its bandwidth rate its bandwidth is specified as the range of 2 to 20 megabits per second. And not only that, MPEG-1 was mostly a standard for storage whereas MPEG-2 since it addressed the high definition television it addressed the communication aspect also. So this also contained the communication standards for the multimedia so that was in the MPEG and called as MPEG-2 and there was another group called as the international called as the International Telecommunication Union ITU. In fact ITU was formed out of the earlier group which was called as Consultative Committee of International Telephones and Telegraphs CCITT.

Now, few years back CCITT was renamed as ITU called as the International Telecommunication Union and International Telecommunication Union they came with different standards for video coding, they also had separate standards for the speech coding, audio coding and all these things so everything and for video also they had come up with some standards.

Now, in the video coding their first standard in the list was the H dot 261. H dot 261 was the standard which addressed the video conferencing applications and it is primarily an ISDN video conferencing application that is the H dot 261. And here the target bit rate is called as the p by 64 kilobits per seconds. So it is a quite a low end of bit rate application; p is an integer and if you

take p is equal to 6 as a very typical figure you reach 384 kilobits per second. So 384 kilobits per second for video conferencing was okay with the least lines and one used the least lines 384 kilobits per second was okay.

(Refer Slide Time: 55:25)




Now they also started the work for the next generation of the standard and this should have been H dot 262. But what happened was that when the MPEG-2 work began then it was found that more or less whatever H dot 262 wanted to address was also getting addressed by the MPEG-2 standards so there was no standard which was formally drafted as the H dot 262 and instead all the H dot 262s charter got into MPEG-2 and then there was also a thinking of having a standard called MPEG-3 but by then the ITU they thought of the standard H dot 263. Then it is this group the MPEG group whose thought that it is better to give the ball now to the ITU so no MPEG-3 was there but instead H dot 263 was adopted.

Now H dot 263 is for very low bit rate applications so it is very low bit rate video coding and by very low bit rate we mean bandwidth less than 64 kilobits per second so it is a highly challenging one and then there was the next standard which is H dot 264 this is the most recent of the ITU standards and H dot 264 is significantly H dot 264 significantly out performs the earlier standards that is H dot 263 and in fact this has got several features several very nice features

which I can tell you in the next class because I need not have to tell you about all the previous standards.

But then, in the MPEG community there was another standard which was MPEG-4 and MPEG-4 addressed the object oriented object oriented audio and video coding. So basically, in the MPEG-4 standard every video stream and every audio stream or every part of part of a video or part of an audio they are referred to as the Audio Visual Objects. So yet evolved the concept of Audio Visual Objects in short form called as the AVO and AVO basically is the main key behind this MPEG-4 standard then there are further MPEG standards which is getting proposed now.

Now, the work is on the MPEG-7 and on the MPEG-21. Of course they address different aspects for which we can make a brief mention little later; in any one of the subsequent lectures I can make a mention about what MPEG-7 and 21 are going to do. So okay, for today we stop at this point and we are going to summarize our conclusion; discuss little bit about H dot 264 and summarize our lecture on the H dot on the video coding search, thank you.