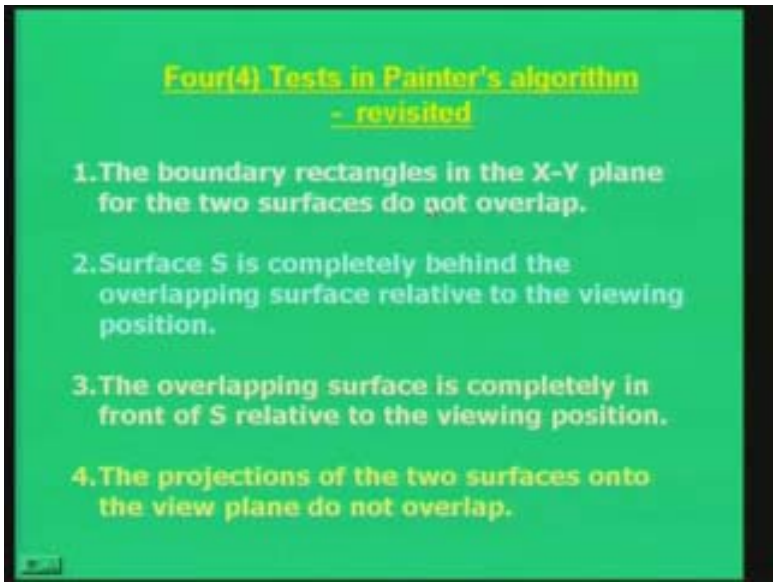
Computer Graphics
Prof. Sukhendu Das
Dept. of Computer Science and Engineering
Indian Institute of Technology, Madras
Lecture #30
Visible Surface Detection (Contd…)

We continue the discussion on Visible Surface Detection algorithms or VSD as it is called. Of course there is another name Hidden Surface Elimination as well. So, we have so far discussed the concept based on back face culling where approximately 50 percent of the faces are removed that was at the beginning of this section of discussion on VSD algorithms. And we basically look at the Z component of the normal vector to the surface and the sign of that basically decides whether it is a back face or a front face. Then among the various VSD algorithms we first discussed the depth buffer or Z buffer algorithm first. Then of course we also discussed the scan conversion algorithm and of course in the last class we discussed the depth sorting algorithm or what is known as the Painter's algorithm.

If you remember, in the last class we ended the lecture with an example which we will revisit shortly very briefly again today to ensure continuity that what are the four tests of Painter's algorithm and then we will look at the examples which you have seen and a few more example today about special cases where Painter algorithms have to be modified or special creaters to be taken for certain distributions of surfaces or polygons which have to be rendered. So we look back into the slide these are the four tests in Painter's algorithm when we revisit them. The first test is of course the Z extent which I termed as the zeroth test does not overlap. Then the first test says that the bounding rectangles in the X-Y plane of the two surfaces do not overlap. So look at the rectangles and look at the extents along X and Y directions.

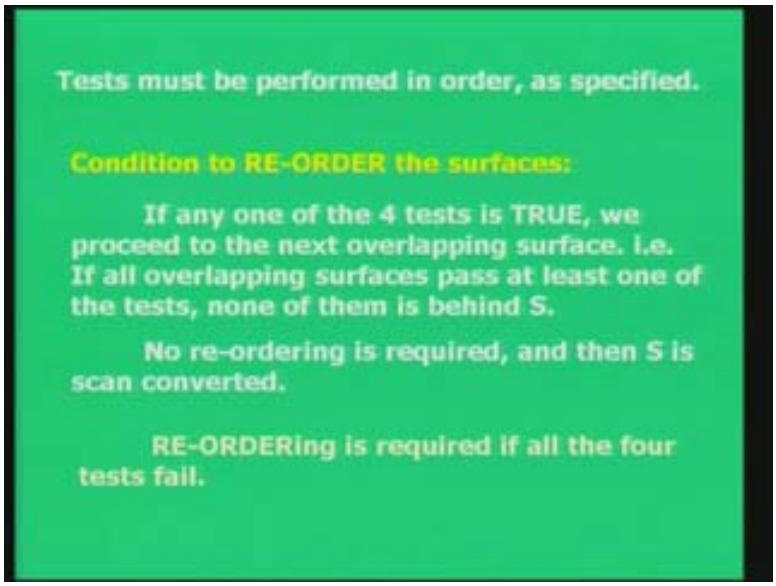
(Refer Slide Time 03:00)



If that fails you check whether the surface S is completely behind the overlapping surface relative to the viewing position because we are going to paint S and you must check whether it is completely behind and overlapping surface in front of it. And if the test two fails then we go to test three which says that the overlapping surface is completely in front of S which is going to be rendered relative to the viewing position. And again the test number three fails we go to test number four which says that the projections of the two surfaces onto the view plane also do not overlap. This test number four is close to one in the sense that one was comparing the bounding rectangles only and the test number four ensures whether the projections actually overlap on the view plane and also. So these were the four tests in order we have to keep checking and if anyone of these four tests pass then we can go ahead and render the surface S with respect to all other surfaces.

So we go back to the last viewpoints, the test must be performed in order as specified already and the condition to reorder the surface is that if any one of the four tests is true I repeat if any one of the four tests is true we proceed to the next overlapping surface that is if all overlapping surfaces pass at least one of the tests none of them is behind the surface S which must be rendered now.

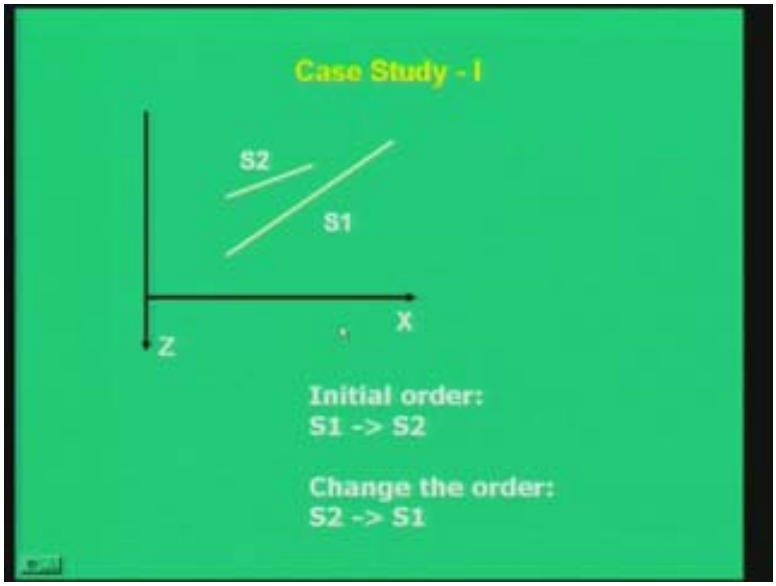
(Refer Slide Time 04:58)



No reordering is required and then S is scan converted if any one of the test is true however reordering is required if all the four tests fails. So we must keep this in mind that any failure of the depth extent overlap and then the four tests if any one of them fail reordering is manageable.
So we look at the case studies as examples of Painter's algorithm once again. We had seen a couple of them in the last class. We will quickly revisit it again, this was the case when the initial order S1 and S2 and we can know that this says that S1 has more depth than S2 as far as the viewing direction is concerned, negative Z direction is viewing direction so from top to bottom is what you are looking and the order hence will be S1 and S2. So here let us pass through the tests where we first see that the Z extents overlap in the zeroth one then the first test says that X and Y

extents also overlap in S1 S2 then whether it says that S2 is completely in front and behind so that test also fails and the polygons also overlap.

(Refer Slide Time 05:53)



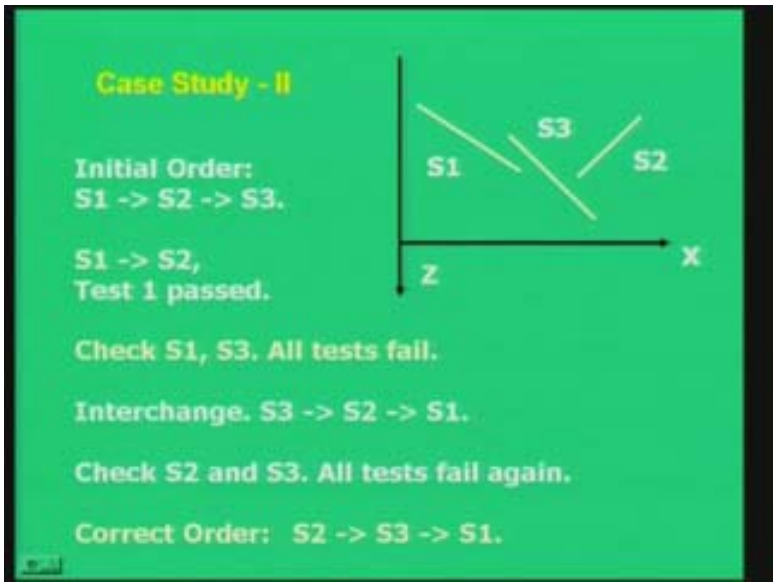
So what you have to do is reorder and change the order to next S1 because all the tests fail in this case. That was the simplest example which we took in the last class and I hope you have understood it. Why all the tests fail in this particular case specially the test two and three are very important to visualize that the surface one must be completely in front, the overlapping surface and the current surface must be completely behind. Both these tests fail as well as in this case. Of course the visualization of the other overlapping X and Y extent and even the fourth test of polygons overlapping is very clear from this particular example.

Let us go to the second case, the case study two where we have three surfaces in this order and we have an initial order S1, S2, S3 Y because S1 has the maximum depth in terms of surface limits then S2 and then S3 so that is the order. So let us pass through the first test S1 and S2 there is absolutely no overlap in fact although the test zero fails but the test one will be passed because the bounding rectangles of S1 and S2 will not overlap so S1 and S2 in that case can be ordered and order can be kept and it could be drawn. In fact you can draw it in any order either S1 or S2 or S2 or S1 we will see that later on. So check S1 and S3 to ensure you can draw S1 and then S3 all the tests fail, why all the tests fail?

You can see for yourself there is an overlap in the Z extent, there is a overlap in the X extent or the Y extent also could be and then neither you can see that if you want to draw S1 and then S3 the completely front and behind test also will be failing because if you want to draw S1 and then S3 which typically assume that S1 is completely behind the S3 and similarly S3 should be completely in front of S1 which is also not true. If you extrapolate these lines we will see that they intersect and hence the case, Then you will find that S1 and S3 all tests will be failing and even the fourth test also will fail so you need to reorder.

So exchange S1 and S3 and then you will find that this is the new order which you will use. In the new order you will find that S2 and S3 now must be compared to ensure that can we draw S3 first and then can we draw S2.

(Refer Slide Time 09:02)



Check for S2 and S3 which are the polygons here all the tests will fail again. Why? The Z extents overlap, the X and Y extends overlap and since you are planning to doise two and then S3 we expected S3 will be completely in front of S2 which is not and the similarly since you have drawing S2 and S3 is completely behind test also fails all the test fails once again and hence S3 is not completely behind S2 and similarly S2 is not in front of S3 and so both the tests two and three will fail test number four also will fail because the polygons do overlap so you need a further reorder.

So when you further reorder the correct reorder what you will get now is S2, S3, and S1. Now you will see that when you draw S2 and then on S3 one of the tests will pass the front and behind and then you can draw S2 and S3 and S3 and S1 has already been reordered so you can draw in this correct order. This is how an algorithm proceeds even if you have more than three polygons. If you have n number of polygons n is a very larger number I keep repeating it is difficult to visualize but please visualize that not only you have ten or hundred polygons but maybe a million polygons.

But what will happen is you start with some ordering of these n polygons based on the maximum Z extent. Then what you have to do is start with the first polygon because it is an order based on the maximum Z extent so can you draw the first one. These has to be compared with the remaining list and if you feel that the test for the first polygon with the remaining which is basically n minus 1 comparison so the complexity of this algorithm will typically go to order n square complexity where typically what will happen is n is the number of polygons. So you can almost visualize that is the level of complexity which we are talking about and in fact order n square amount of comparisons is what you have to make with one polygon n minus 1 and

similarly for the other n minus 1 polygons also you have to do that. And what you do with first polygon and the same thing you do with other polygons is you need to check whether rest of the n minus 1 polygons can be drawn after you draw the first polygon.

Can you draw the first polygon before the reverse it means that can you draw the first polygon before you draw the rest of the n minus 1? And to do that you have to compare the first polygon with rest n minus 1 to ensure that you draw it first and one of the tests at least must be passed when you are comparing the first polygon with any of the remaining tests. If that is not the case you need to swap and then restart the checking. That means you are basically looking among the n polygons as to which is the first polygon. The worst case complexity can even actually go to even higher order n queue. But what you are basically trying to find out is which of these n polygons can you draw first there must be one polygon which are all front facing of course forget the back facing polygons because back face culling is already been used, that is what the basic assumption we made for all VSD algorithms.

Now after back face culling is done in the case of Painter's algorithm we are trying to find out which of these polygons can be drawn first in terms of which is having the greatest abilities behind everything else. All the polygon can be drawn after that whether it is two or three or four or higher or ten or hundred or thousand or million polygons which of these out of those n polygons can you draw first. That is what you need to find out. You could be lucky if the initial ordering is based on the maximum extent of the depth works out that the first polygon can be drawn first. But it may not be the case in a very complex scenario. So what you need to do is to reorder if necessary if the tests are failing, keep on rendering all of them one by one and definitely there will be one particular case where you will find a polygon which could be drawn first. Once that is ordered that is flagged down and then you work with remaining n minus 1 and keep checking the same sort of a thing. So the computation work case basically you can go to order n queue but typically order n square and average case complexity what you can estimate for the Painter's algorithm n being the number of polygons which have to be rendered. This is the key idea.

I hope I have giving visualization only with two or three you can visualize if I take higher and higher value of n as an example view you can actually workout I leave it as an example to combine case studies both one and two and come up with four surfaces and check it out. But you can see the number of checking have to be manually done to understand the methodology or the way the algorithm was that is the key important point. Since we are assuming for the time being that with the help of case studies one and two you are able to understand the algorithm now and we will say some special cases where some precautions have to be taken in the case of Painter's algorithm where things could be very difficult. Let us take this example, you are taking this example where we are in the process of checking the order which could result in an infinite loop. How will the infinite loop come? There are three rectangular polygonal strips because it is polygonal and all of them have four vertices that is why it is a rectangular strip in 3D and one is overlapping the other one in a sort of a cyclic fashion.

As you can see that in some parts the red is above the greenish patch and the greenish patch is above the whitish one and similarly the whitish one is also above the reddish patch. Now you start with some Z ordering where there are three polygons S1 S2 and S3 there are three of these.

I did not level them but you can level them but yourself in your notebook. So based on some maximum Zth depth extent you will get some order assume you get some order S1 and S2 S3 it could be red first and then green then white or white, green and red whatever order you pick up does not matter. You will find that when you are trying to check especially the test two and three because zero which one is completely in front or back and you will be forced to push one back and cyclically you will come back again.

Let us say you start with S1 S2 S3 you will find that you will reorder S2 and S1 then you will start comparing with S3. After sometime you will find that again you have to reorder S1 S2 back because all tests are only failing. In fact you will not be able to find the surface in this case I just talked about this point earlier because if you have n polygons and after you arrange them in descending or ascending order of depth depending upon the view direction of course you need to find out the surface which you can render first or shade first among the list in this case n is equal to 3.

If you look in to the figure again you try to visualize which of the surfaces can be drawn first. Now you see here in this case it is none because either you try to draw the red strip or the green one or the white one. If you draw any one of them first what will happen is the other two will completely overlap the other one which is virtually drawn next and that is never the case because one polygon is overlapping to some extent and it will be overlapped by other one. As for example if you draw the red strip first what will happen is not only the white the green strip which is also drawn second will also overlap the red one. So you will not be able to find out once such surface which can be drawn before the other two and this is a case what we call as the reordering entering into an infinite loop.

I leave it as an exercise to check that infinite loop for yourself by taking in any order and comparing between pair of these rectangular strips and checking all the four tests and reordering again as we have done in the case of the previous case study. Case study two we took, remember, three surfaces S1 S2 S3 and we reordered and we finally got a sequence which passes all these tests or at least one of those and the we got the correct ordering. Here what will happen is you keep on reordering and keep on reordering and will never be able to find out the correct order because the physical significance of this is you can look at the figure once again and see for yourself that you will not be able to draw a single surface and then draw the other two in any order. The other two does not matter in whatever order you draw. If you check any one of these three and draw it first you will not be able to get this correct impression of this picture which you have shown in terms of partial overlaps.

Look into the figure again you can take the green one, the white one, red one etc, and if you draw any one as the first polygon to rendered the other two will completely overlap that and you will not get the correct result. So this is the physical significance of when checking the order results in infinite loop you will never be able to come out I again repeat with the single polygon in this case which can be rendered first. Let us look at the array example with just two polygons it is also the same case. As you see here both oval polygons overlap the other one, the red overlaps the green portion at some point and the blue one also overlaps the red.

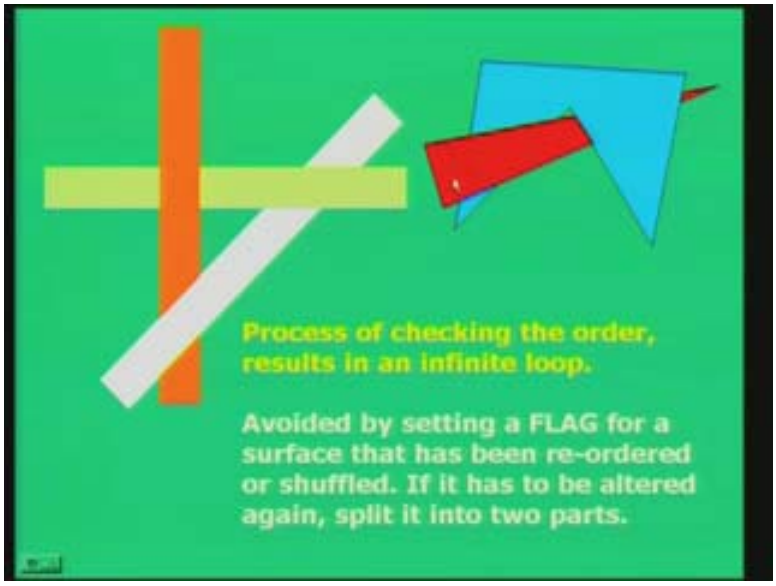
What will happen is, you start with some order absolutely no problem and then you always keep on reordering and that will result in a problem that you will never be able to find out the surface, even these two cases can be drawn because if you draw the red one and then the blue the blue will completely overlap the red and the other case is, if you draw the blue and then draw the red the triangular patch in the red one will completely overlap the blue color polygon. So these are the cases where you cannot draw any surface at the first and then draw the other or the remaining one. This process of reordering of Painter's algorithm fails in this case because you entering into an infinite loop and you keep on reordering based on certain constraints and you will never be able to stop.

Let us look at the solution. This case can be avoided by setting a flag for a surface that has already been reordered or shuffled. That means you start with the case of two or three polygons and suppose you have reordered any pair. If the process of reordering, when process of checking other polygons if test is to be re-altered altered again you do not do that if a flag is set it must be split in two parts. So, splitting a polygon is necessary in this case for Painter's algorithm whenever there is a scope for you to enter into an infinite loop.

How will you enter in infinite loop? Let us say the case of that red triangular and the blue polygonal patch S1 S2 you start with some order. All the tests fail and you reorder them basically. So you can land up with S2 S1. Check for S2 S1 to see if the tests are passing, you will find all the other tests fail and then again go back to S1 S2 and that process will continue cyclically and that is the case of the infinite loop and the physical significance also has been discussed which means that you cannot draw any polygon before you draw the other or the remaining one. So, how to avoid that? Set a flag, what does it mean? When you start with some order before checking the test S1 S2 and you have swapped them or reordered them. And when you do that you set a flag that the S1 S2 pair has been ordered already once and the process of checking either with only these two or the remaining polygons together in case of three or more such cases if it requests for another swap S1 S2 which was already swapped again if you want to redo that operation that means going back to the original position no please stop that because you will check the flag and find out that these two polygons have been swapped.

What is the solution? You have to split one of these S1 and S2 into two parts. That means there are two polygons set as S1 a polygon here and other polygon S2 here and you keep on reordering because you cannot draw any one of them but what you do is you take a plane intersection of one of these with the other one and split one polygon into two parts, any one of them not both because both may not be necessary. You can do that but that will be increase the computational burden. You take one polygon. Remember, a plane can intersect another plane similar to a clipping problem which you know so it is clipping with respect to a surface and the two intersections of two polygons or surfaces within the line is used to split the polygon in two parts.

(Refer Slide Time 21:36)



So from two you result in three polygons and we will see. I leave it as an exercise for you to draw these cases of overlapping triangles or rectangular strips where if you split any one or maybe two in certain cases you will able to order the surfaces correctly and one of the test will definitely pass. So I repeat again for the second case where you have this red triangular patch and the blue polygon you could split the red one into two parts, the one which is visible to you and other in the back side.

Now you can first draw this smaller part of the triangle the red one which is behind then the blue and then the red one. That will be the order which will be resulting in this. What you do in this case let us say the red one can be split into two parts. So you draw the smaller strip and then finally the white one also could be split into two parts draw the smaller white strip then you can draw the full green one then the white one and then the second red part. In this case of three polygons the rectangular strip it is necessary to split two of those polygons. In the cases of two overlapping you need to split only one of them and you get the ordering. Remember, you need to set a flag so that you do not reorder and enter into a scope of an infinite loop you need to stop that provision and so what you need is basically a provision.

Again I repeat; to set the flag, stop the reordering because if you do not set the flag and enter reordering more you are going into a infinite loop nobody can stop you, your program will never terminate and so you stop that process entering into an infinite loop by setting a flag and say please I am not willing to reorder it once again and I will go for splitting. So take those polygons and split any one of them and that is the provision which you know mathematically how to clip one plane with respect to other one. Once we know that, that is the particular case of splitting polygons in the case of Painter's algorithm.

What happens in this case? This is also very nice example, you see here that there are two rectangular planar patches strips again and S1 and S2, S1 is in red color and S2 is in blue color.

What are these values given? You are looking through the viewing plane X-Y looking along negative Z direction let us say and the values which are given at the vertices of this rectangular strips or polygons henceforth I must call it rectangular strips in this case are the Z values of the vertices. XY does not matter whatever they maybe these are the Z values. So, if you look at the vertices of S2 in fact the surface S2 is parallel to the X-Y plane because the vertices are at 0 the Z equal to 0 so it is a planar pack which is parallel to X-Y plane. It is not the case first surface S1 why? The bottom two vertices have depth plus 0.5 each and the top two vertices have depth minus 0.5 each. So you have to visualize this scenario that all those surface S1 and surface S2 appear to be like this in this form one is of course a red color I will put two colors let us say like this where you have two different colors for surface S1 S2 although they are visually appearing but one of them is just tilted in this form.

Surface S1 is this polygon which is parallel to your X-Y plane your viewing plane in the screen which you are viewing this lecture right now which is the X-Y plane and this is parallel to that X-Y plane this is a surface S1. S2 is not parallel it is inclined. It is inclined in the manner like that and that is what you have for S2. So you probably view it is be like this not like this. The two planes S1 S2 if you see are not parallel to each other one of them is inclined. If this is Z direction X-Y plane is like this one of them is X-Y plane S1 and the other S2 is like this so this is what you have to visualize.

Look again and try to visualize these two surfaces S1 S2. Are you able to visualize now? Difficult but try do visualize. Again I repeat S2 is parallel to X-Y plane and S1 is not parallel to X-Y plane look at the value of the set coordinates of the vertices given for surface S1 S2 next to the vertices. I repeat, 0 is the Z coordinate or depth of the vertices of the polygon S2 whereas for the rectangular strip S1 the values are 0.5 for the bottom row two vertices the upper pair of vertices have a negative value of minus 0.5 so S1 is inclined.

Let us go through all the tests here for this particular case. What about the zeroth test Z extents overlap yes they do because as you see here this is S1 S2 here so if you look this is the Z direction so Z extents overlap zeroth test fails so go to the test number one. Test number one says bounding rectangles do the overlap no. If you take the bounding rectangles in this case S1 S2 each of the rectangles itself is a bounding rectangle. So we will assume that although there could be a common boundary between the two we can neglect that to be an overlap. So test number one itself is passed and we can render S1 S2 in any order in fact the ordering now will be S1 and then S2 why? It is because that the extents of Z which was used to put S1 and S2.

Look back again S1 and then S2 because S1 the prior ordering before the test will S1 and S2 so you can render it in order no need to reorder them. Now look what I do with one of these polygons in this second figure B, this was figure A, look at figure B carefully and see the variations with respect to figure B. Look what we have done. We have kept the polygon Z extents same in terms of the Z coordinates of the vertices that is kept same for both surfaces S1 and S2. In this case of course it is the same, the rectangular strips are used S1 S2 Z extents remain the same, I rotate surfaces S2 in the X-Y plane about an angle theta. You have to visualize this. How do you do? Let us look at it. This was surface S1 and you have a surface S2 which is inclined, surface S2 and this was S1 this is surface S2 which is inclined.

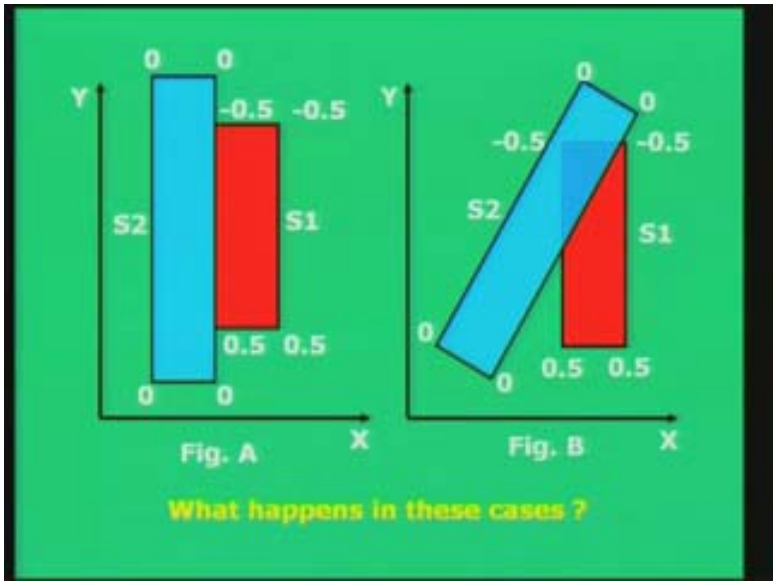
What I am doing basically is taking surface S1 and rotating in the X-Y plane. I can do that because there will not be any problem of two surfaces intersecting each other in this case, it will not happen. I repeat, S1 is parallel to X-Y plane and you can rotate that in the X-Y plane. And you are doing that with the surface S2 inclined in the Z direction but it is adjacent only to your surface S1 this is S2. So S1 is now rotated you can do that. This is my palm let us say S1 and it is rotated with respect to S2.

Look at the figure once again and see how you can visualize this. From figure A to figure B you can go by simply rotating the surface S2 in the X-Y plane and it will still not intersect, well there will be overlapping in the XY extents but it will not cut or intersect or getting to the surface S1. The surface S1 and S1 are not intersecting which is the scale. It is like around the vertex of meeting point between the two common lines S1 S2, I have given it a small rotational twist and that is S2. And the top part of S2 is now overlapping S1 and the bottom part has been separated out.

I hope you are able to visualize this figure B and the Z extents of the rectangular strip, there is absolutely no change because for S1 this is the same, I have kept it undisturbed or unaltered it is only for surface S2 I have given a rotation about the X-Y plane and we know when we rotate about the X-Y plane the Z coordinates in 3D do not change. That is what is happening with the surface S1 that the Z coordinates are remaining the same you have just given it a twist, the other surface S2 is unaltered.

Look back now let us see the test what happens. As you can see for figure B that the zeroth test fail because like in figure A the Z extents where overlapping so test number zero fails. What about test number one? Now the polygonal boundaries, the minimum enclosed rectangle for surface S2 will get inside the enclosed rectangle or the rectangle itself for S1. Remember, the minimum enclosed rectangle for surface S2 will not be S2 anymore. It will be a larger rectangle which will get inside S1 or S1 get inside the minimum enclosed rectangle for surface S2 so what happens is test number one also fails.

(Refer Slide Time 29:23)



What about test two and three? Well if you want to order S1 and S2 you have to check whether S1 is completely behind S2 and S2 is completely in front of S1 neither of these cases happens. You have to visualize in 3D now if this is S1 this is your S2 and I have rotated it a little bit for figure B.

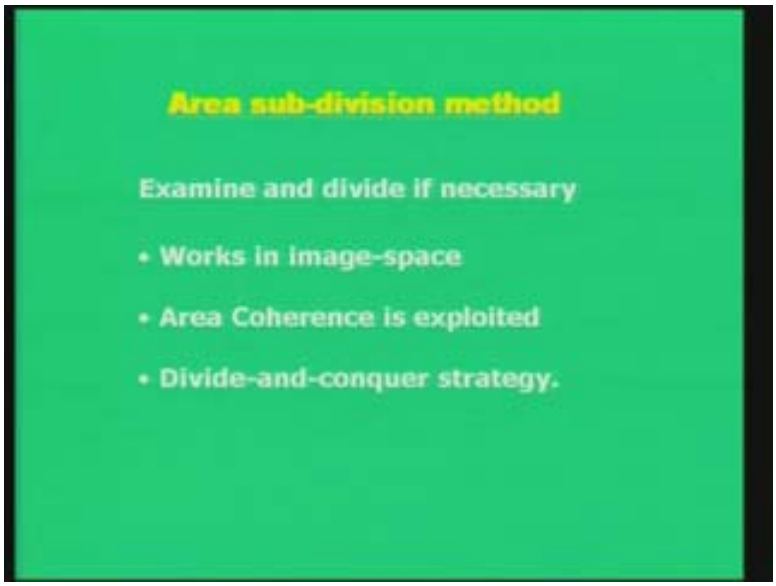
You can see for yourself if this is the Z direction what will happen basically is that the front and behind test also fails. None of them is completely behind the other one the other one is not completely in front of the figure S2. So S1 S2 test number two and three also fails. The only remaining test now is test number four. What does test number four say? You have to project these two polygons on the X-Y plane. As you go back to the figure now the polygons themselves overlap so for figure B the test number four fails as like test number one also. Not only have the bounding rectangles overlapped but the polygons themselves overlap so all the four tests have failed for figure B.

And so what do you do? What you do for figure B? For figure B there is no other option as we have did in the previous cases we have to split one of the polygons. We have to split one of the polygons in two parts use S1 to split S2 in two parts or you can use S2 to even split S1 into two parts. I did say when there are two polygons the reordering will take place returning into an infinite loop we set a flag and say we reorder only once but not two times avoid that.

And this case the second one is demanded when recheck of all the tests for four tests fail then you need to split the polygons into two parts and that is what you do for figure B. So what you do here? I repeat, use S2 to split S1 into two parts or you can take S1 and split S2 into two parts you will be having three polygons now and you will be able to come with a correct order. I leave it as an exercise for you to take S1 and split it into two parts or even S2 and check the reordering and the testing once again.

If you split the red rectangle which is surface S1 into two parts you can draw the back side of S1 first then the blue one and then the remaining part of the S1. That will be the order which you will get in this particular case. I repeat again; if you split S1 into two parts the back side of S1 first then the S2 inclined blue surface and then the front remaining part of S1 is what you can draw and that will be the correct ordering and the correct impression of this figure which you will get. So you can only implement that by splitting. That comes to the end of the Painter's algorithm which was the third VSD algorithms which we have discussed so far. We move on to a next concept called Area sub-division method.

(Refer Slide Time 33:24)



Area subdivision method is the fourth category of the VSD algorithms which we are going to discuss. And the concept says, examine the polygon and divide if necessary. So this goes straight away into the concept of division which was a special case of Painter's algorithm. In the case of area sub-division we examine and divide if necessary. The other key points in area sub-division methods are; it works in an image space. What you mean by image space here? You remember both in Z buffer algorithm or depth sorting Painter's algorithm or even scanline of course this scanline of a mixture of both but basically depth sorting painters as well as the depth buffer we are working in the object based polygon by polygon and we are only involved in the depth value surface normal concept. That will also come but here we are working in image space.

We will do manipulations of projections of the 3D polygons on to 2D and then work in the image space. Of course you have the depth values because that is the essence of VSD algorithm in terms of Z ordering. We cannot avoid that but the methodology of working in the area sub-division method works in the image space rather than what we called as object space in the other algorithms.

You remember, one of beginning lecture of VSD I did say that I can have two broad categories of VSD algorithms one is object space and image space. So far, most of the discussion was on object space, this is the first time except the scanline algorithm is probably close to the image

space all these combinations of both but it probably can be visualization of image space. Area sub-division in fact is a good example of image space based method by which we render polygons and implement VSD algorithms. So come back, we are working in the image space. As since we are working with image space with polygons area coherence is exploited and we divide and conquer because we did say earlier that we have to examine and divide if necessary. So it is basically a divide and conquer strategy area sub-division method. And the algorithm which we are going to discuss of course there are quite a few. We will discuss the simplest one which is known as the Warnock's algorithm for area sub-division method and we will see how it works.

We talk in the case of area sub-division method a possible relationship of the area of interest, a rectangular area of interest AOI this means the area of interest, nothing but the area of interest AOI which is a typical terminology used in computer graphics and the imaging community please remember this area of interest. So we are talking of possible relationships of the area of interest which is a rectangular area of interest and the projection of the polygon. So we are here to compare rectangular area of interest.

We will see rectangular area of interest but for the timing you can visualize that it could be the entire image because the image itself is a rectangular area and the entire area of interest could be image itself or image itself could be an entire area of interest. That is what we are comparing with a polygon. Or you can take a small part of an image, if this is an entire image which you are taking now on this screen take a small part of the image which is considered as a sub image by imaging technology people take that small rectangular strip and you are comparing that say a rectangle which you see behind. Or any other polygons which you draw you are comparing a polygon with small rectangular strips.

We will see with illustrate examples next as you move on this course but for the time being assume that you have a small rectangular strip which could be the entire image also or any small part of rectangular strip and you have an arbitrary polygons somewhere with n vertices and you are comparing these two in the image space itself to start with. That is what we are talking about possible relationship of the rectangular AOI or rectangular area of interest and the projection of the polygon in 2D. So everything is going on 2D right now.

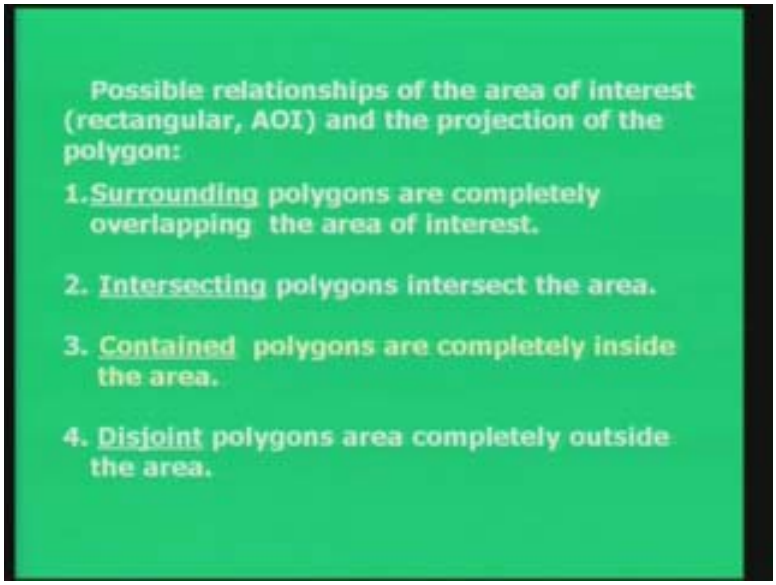
What is done in 2D? Surrounding polygons are completely overlapping the area of interest that could be one case. We will see what you mean by this. We will go through this possible relationship in terms of statements and then go to the example in the next slide. We see this relationship one relationship is surrounding just remember this word surrounding.

We are talking of surrounding polygons which are completely overlapping the area of interest. Remember, this area of interest or AOI is a rectangular sub image or a part of the image under consideration or it could be the full image also to start with. You can have intersecting polygons which intersect the area. That could be another case. We can also have a contained relationship where we say that the contained polygons are completely inside the area and the fourth part is disjoint polygons where the disjoint polygons is a case where the area is completely outside the area of interest. The polygonal area is completely outside the area of interest. So these are the four different relationships which we between the rectangular area of the image any arbitrary it is not completely arbitrary we will see how to choose that rectangular area of interest which will be

the key part of this area sub-division or Warnock's algorithm a small rectangular strip is what you choose you have an arbitrary polygon somewhere and you are talking of a polygon let us say here and a small rectangular strip. There could be various relationships and we are seen that the only four of these which cover all possible relationships.

And remember, these four terms which describe this relationship between rectangular area on the polygon. The first is surrounding, intersecting, contained and disjoint. I repeat, surrounding intersecting contained and disjoint. Remember these four terms, please note it down, these are the four relationships which can exist between these two sub-areas within the image plane, one is a projection of a polygon in an image and a small rectangular area of interest or sub image or a part of the image. So we look into examples which illustrate these relationships.

(Refer Slide Time 37:56)

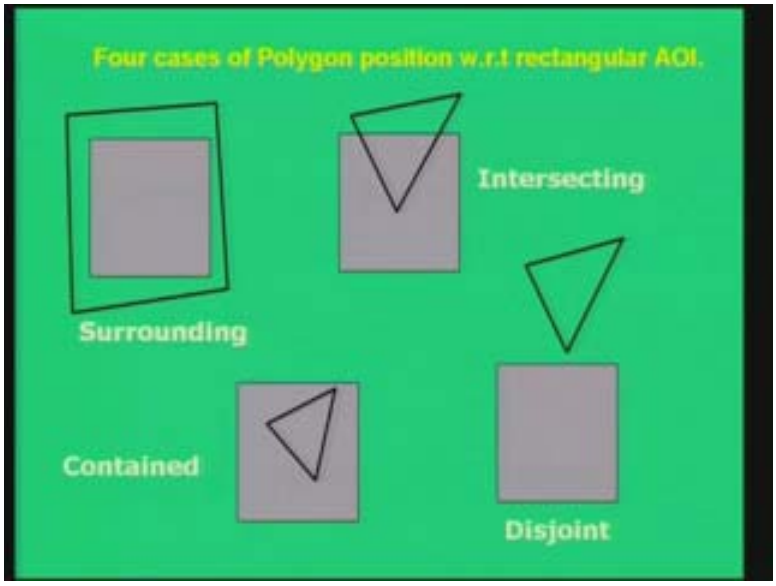


Let us look at the first example. So we look into the four cases of polygon position with respect to the rectangular area of interest. The rectangular area of interest is given by this grey color rectangle which could be a part of an image or the full image. This is an arbitrary quadrilateral which I have drawn which is the projection of the polygon. And as you can see that the polygon is completely surrounding the rectangular area of interest which is this grey color strip which is not the polygon projection.

Remember, you must visualize that this rectangular strip is the area of interest which is the part of the image which you are analyzing or considering. And we are comparing this rectangular portion of the image with an arbitrary polygon and this shows an example where this polygon completely surrounds the rectangular area of interest and that is the case of what we call as the surrounding relationship. This is the case of surrounding relationship. I hope you are able to visualize this example. Of course when we probably see other examples this will be clearer. This is the case of intersecting.

As you can see the projection of the polygon in this case which is triangle now from a quadrilateral I have gone to a triangular polygon and so projection of triangle will be triangle in 2D. And we can see partly this position of this triangular polygon will be overlapping with this rectangular area of interest which is this grey color rectangle.

(Refer Slide Time 41:27)



So this is an example which illustrates the case of intersecting a triangle with this rectangular area of interest. So that is the case of intersecting. Surrounding intersecting two mode of contained and disjoint. This is contained, the same triangle now of the polygon is completely within or inside the rectangular area of interest so that is an example which illustrates the concept of contained relationship and the last one is disjoint where there is no overlap area between the triangle and the rectangular area of interest.

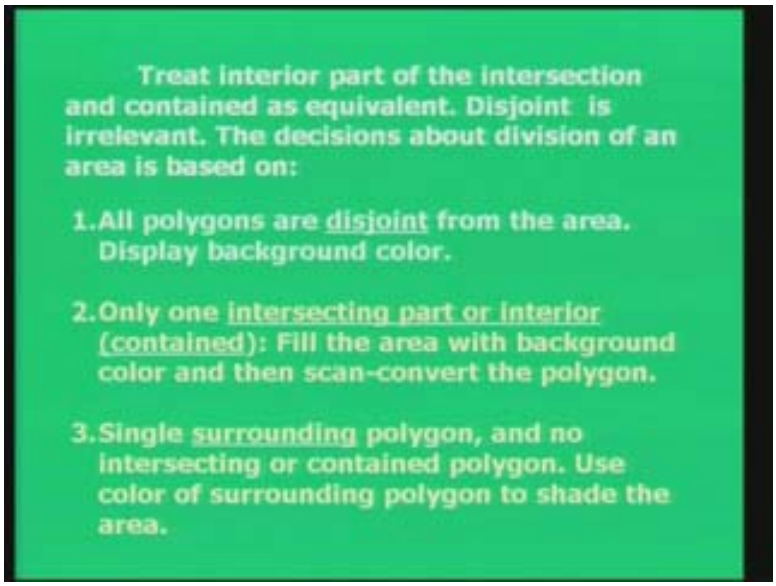
I hope these four figures are able to illustrate the four possible or four different cases of relationships of the areas in 2D between the projection of a polygon and rectangular area of interest. I repeat, please remember these terms surrounding, intersecting, contained and disjoint. I repeat surrounding, intersecting, contained and disjoint. If you remember these four relationships and think of these illustrations which you have just seen in the previous slide that is enough for you to go into the steps of the algorithm and visualize what is going to happen. Please look into the slide once again. These are the four cases of relationship surrounding on the left, intersecting on the top right, contained in the left bottom and disjoint on the lower right.

So I hope you have understood this and also copyedit in your notes. Now we will look into the case of how you treat these four different cases. Treat the interior part of the intersection and contained as equivalent. That means hence forth intersection case and contained will be treated as similar or same in terms of interpretation of rendering in terms of VSD. What we will do? I repeat again, intersection and contained how they are equivalent, why they are equivalent in terms of the algorithm which will be clear but for the time being let us visualize not in terms of the relationship that they are similar, no, contained is fully inside intersecting and is partly

overlapping mind you. But in terms of interpretation of the algorithm steps we will say whenever we have a relationship which is either contained or relationship which is intersecting, interesting and contained remember that they we will be treated as similar. That is the first step.

We do not worry about the disjoint case at all, it is irrelevant. Whenever we find the disjoint case we will not worry about that polygon at all. And the decision about division of an area, remember we are talking of an algorithm which is based on area sub-division so you have to divide in area is the question. When do you divide an area? That is the criteria which we will talk about. Look back, the decisions about the divisions of an area is based on the following constraints.

(Refer Slide Time 46:28)



If all polygons are disjoint from the area display the background color do not subdivide that is what we meant by saying sometime back that the disjoint is irrelevant that means just neglect that polygons if it is disjoint and display with the background color. If you visualize a background color like a grey color or a green color which we are seeing on the screen it could be a dark black color or the background also could be white use that background color if the polygon and that area of interest at disjoint. That means completely and absolutely there is no overlap. So that disjoint case is irrelevant and all polygons are disjoint from the area, display the background color. Point number two says that if there is only one intersecting part or interior which is contained.

Remember, the first line or the first paragraph says the interior part of the intersection and contained are equivalent that is how you treat them, this is the second point which is related to that first statement. If you have only one intersection part or interior or contained so these two relationships are same intersecting and contained and if that is the case fill the area first with background color and then scan convert the polygon.

If you have drawn those pictures which you have seen in the previous slide we talk about these four illustrative examples of the relationships between rectangular area and a polygon if you know that already. If you know that already you know that in the case of a polygon contained in a rectangular strip or intersecting one if I draw the background color first of the rectangular AOI or the area of interest and then shade the polygon of course within that particular area only do not draw the complete polygon, so you have to clip and find the common intersection part if it is intersecting and scan convert only that part and if you do that you will get the correct impression of what we have seen in the previous figure.

So I repeat again; first point is very simple, throw away disjoint part, display background color only and when we are intersecting and contained fill the area with background color and then scan convert the polygon. So one and two steps have taken care of these three relationships and the only one relationship which is remaining now to consider is we discussed about disjoint, contained, intersecting all the three and which is remaining? Overlapping relationship is the one which is remaining between the two. What you do when you have an overlapping case? If you have a single surrounding polygon and no intersecting or contained polygon you need use the color of the surrounding polygon to shade the area that is very simple. So the subdivision as not yet come we are looking into scenarios with which we can handle without sub-division of an area this is a simple case.

So we have discussed about disjoint earlier, contained, intersection under one category surrounding if you have just single polygon completely overlapping a surrounding rectangle area of interest there is no polygon in front of it or on back side whatever the relationships of other polygons if those are not there at all there is just a single polygon which is surrounding that then you can take the area of interest and display the color of the polygon, use the color of the polygon to shade that corresponding rectangle only.

So these are the cases which you can keep in mind in terms of the decision to be taken based on the relationship between rectangular area of interest and polygon. Still there are so many other cases which are possible which will cause subdivision. So any one of these cases occur in single polygonal cases you go ahead without subdivision but there are two you have to examine for that and subdivide if necessary and that is what we will see.

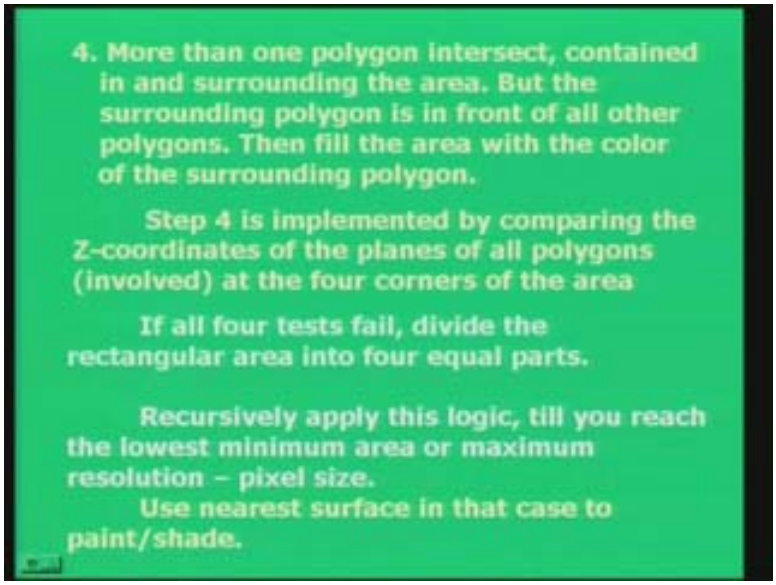
This is the fourth the most important point, if there are more than one polygon which could be intersecting, contained in and surrounding, we are not worried about disjoint. I repeat again, disjoint of course they are irrelevant we say that earlier we were talking of two or more polygons. If there are more than one polygon which is intersecting the area contained within the area or surrounding the area now you can two of those one intersecting one contained one intersecting one surrounding and all that various relations which are possible two or more and they are combinations of intersecting, contained and surrounding except disjoint. We do not worry about disjoint. So, more than one polygon could be intersecting, contained or surrounding the area. If this is the case, but we have just one surrounding polygon which is in the front of all other polygons then in that case also we can fill the area with the color of the surrounding polygon this is interesting.

We have many of those but still there is one in the front. In fact one in the front of you which is surrounding the entire rectangular area there could be others behind which could be intersecting contained in but there is one surrounding one in the front.

Of course if there more than one surrounding you have to find out which was in the front, that you have to move to the object space to do that. But there is one completely in the front which is surrounding in the entire rectangular area of interest and it is obscuring or occluding all the polygons behind when you are looking towards me let us say the polygon surrounding in the front and there are smaller polygons contained and they are intersecting the back side then that front polygon could all those used to display the rectangular area of interest. So one of these four conditions are not true then you need to subdivide the area into more than one part.

Step four is implemented. We will see how it is implemented. It is done by comparing the Z coordinates of the planes of the polygons involved at the four corners of the area and if all these four steps are not true then of course we need to subdivide. We will do that on a later stage.

(Refer Slide Time 52: 30)



If all these four tests fail divide the rectangular area into four equal parts we know that. Of course you have to worry about step four implementation but remember that all these four tests are not like the Painter's algorithm but before you go to these four tests in the object in the image space check whether we can we render the rectangular area of interest AOI with a background color and then the polygon or the full polygon depending upon contained, intersection or surrounding that is what we are looking and if all these four tests fail we subdivide them into four equal parts. This is an interesting part where of the sub-areas are connected. How do you divide image into four equal parts. And recursively keep on subdividing till you reach what you call as lowest minimum area or maximum resolution of one pixel.

Take an area, take the full image let us say this is the full image and you need to subdivide into four parts. Like four quadrants of the image remember when we talked about Bresenham's line,

ellipse, circle algorithm we talked of four quadrants in 2D space and that is what you do in image and check individual parts and see if one of these four tests are true otherwise you have to divide each of those into four parts. You keep on doing use this example still you might reach a case where you keep on subdividing soon not the entire image but at least a part of it and you reach the lowest what is called the lowest level of decomposition.

You can get actually a tree structure, we talked of trees representation of a tree for an image equivalent to Octree representation for solid modeling you remember that. In case of 3D we had Octrees that was derived from the logic of quad trees in the case of an image. That is what you do and you can further subdivisions and so on and the tree might grow along one particular channel at the maximum level of decomposition of this image as it is called an imaging technology you may reach the level of a pixel resolution.

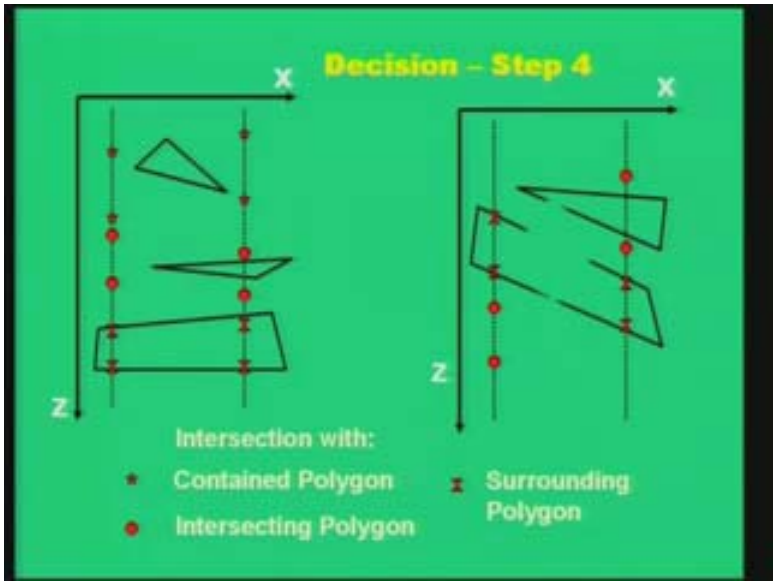
One pixel resolution is what you have this is a common concept which is also used, the three level decomposition not full tree decomposition but sub trees in the case of what is called as wave let the composition but this is outside the scope of this particular lecture. But I can divide an image space into four quadrants and each of these into further sub quadrants and so on. So you can have a quad tree representation of an image space. And you keep doing that and the maximum depth where you can decompose you are reaching the maximum resolution of the image or the minimum quantized step of imaging process which is one pixel of your graphics screen and that is what you see. Recursively apply this logic till you reach the lowest minimum area or maximum resolution size of pixel size and then use the nearest surface in that case to paint or shade because you cannot split a pixel at all so in that case use in nearest surface.

So in that case only you go back to the object space otherwise you are always working in the image space when you are performing this test subdividing again performing this test and subdividing and everything is done in the image space only at rendering time when you finally know what you are rendering you go to object space find out the shade of the polygon and then you use it to render, that is where you go to object space in terms of depth ordering otherwise everything is in image space.

Let us take this example of trying to find out what is this decision in step four. You remember this step four test? We are trying to find out if there is a big surrounding polygon in front of you and it is obscuring or occluding all other polygons in the back side. In front of view a large polygon covering the rectangular area of interest it is in front and all other smaller polygon having the back side. In such a case what you need to do is to ensure that there is no other polygon which will be coming out from the back side within that.

So let us look at this decision test four where you will see that there are three such polygons and we are talking about intersection with respect to rectangular strips of this area of interest you are looking in the ZX plane so you have to visualize these dashed lines which are nothing but lines which are parallel to Z axis and passing through the four vertices of the rectangular strip. And you are taking the projection of ZX plane and when you look here the intersection of the first plane which is the contained polygon with these four lines are marked by, this triangle is the contained one and these four are the intersections of that polygon with these four lines.

(Refer Slide Time 54:30)



Then the circular darks represents the intersection of the intersecting polygon with these four lines and if this surrounding polygon is in the front, in this case you assure that the surrounding polygon is in front because the Z coordinates are at the maximum value and you paint it. But what about the second case? The other case if you see here there is a surrounding polygon but what could happen is there could be a case where that is a intersecting polygon in the back side which is overlapping and coming from behind. It could intersect at some point but there could be something coming from behind which is visible in this case the decision step four could fail if you are looking in the Z extents. Z extents of the intersecting polygon although the intersecting polygon is behind but its Z coordinates of intersections with these fours lines which are parallel to Z axis are in front of the surrounding polygon. So this test may fail, it is not a good test to ensure that the surrounding polygon is in front.

You remember, the figure shows it is in the front but there is a plane behind you and you are looking into some Z extents of some rectangular strip with these intersecting polygons some of the Z extents have come forward but the polygon may not have come front but if you visualize an infinite plane intersecting of the infinite plane with the four lines parallel to Z axis those intersecting points could come out and create a problem in this case you might have to sub divide the polygon. Sub-divide the area of interest and get four parts otherwise that is what you have.

We will stop this example and continue in the next lecture with more cases of area sub-division method. But this is one drawback which you must keep in mind about the decision based on step four based on Z coordinates that is the intersection of the plane of the polygon with a rectangular parallelepiped sitting along the Z axis. So it is something like a viewport in 3D which you are seeing here and you are trying to find out where is the intersection of this polygon with the four extents the four lines parallel to Z axis of this rectangular parallelepiped if the Z extents is in front unfortunately you will not be able to detect the surrounding polygon. But in both these cases you should be able to paint the polygon with the surrounding one. In the second case as you look back in the figure you need to split into four parts in the second case in the first case it

is not a problem because the surrounding polygon Z extents are definitely in front of the other polygons and you do not need to split but in the second case you need to split although actually it may not be so.

So we stop with this example and in the next class we continue the discussion on area sub-division with some more examples, thank you very much.