

Computer Graphics
Prof. Sukhendu Das
Dept. of Computer Science and Engineering
Indian Institute of Technology, Madras
Lecture - 14
Scan Converting Lines, Circles and Ellipses

Hello everybody, welcome again to the lecture on computer graphics. In the last lecture we started to discuss the concepts of a scan line method based line drawing. And we first discussed the difficulties of drawing a line given in the specifications of the starting point and end point of a line and whereas in the continuous case we could draw a line but in the discretized domain we have to fit a set of addressable pixels with respect to the integer x and y coordinates and to do this we have to find out the set of pixels, the integer coordinates of pixels which were closest to the line and we have to do this very fast, as fast as possible so that the computational time and cost is the minimum.

We first look at the incremental algorithm called the DDA algorithm, digital differential analyzer or digital difference analyzer and we had seen that it is a costly algorithm because the steps involved were using a round function and a floating point addition. So to eliminate the round function and also to eliminate the floating point operations and move towards integer based addition which will provide a fast method to produce the xy coordinates of the points which are closest to the line, we moved on to the concept given by Bresenham's which is called as Bresenham's algorithm or mid point line algorithm. And we have discussed the condition of mid point criteria to find out that the mid point of the second or the next point is below or above the line. So we will start from the mid point criteria revisit it again to maintain the continuity and then look at the steps of the algorithm and look at an example as well as to how the points could be generated to draw a line on the screen.

(Refer Slide Time: 4:44)

MIDPOINT LINE ALGORITHM
Incremental Algorithm (Assume first octant)
Given the choice of the current pixel,
which one do we choose next : E or NE?

Equations:

1. $y = (dy/dx) * x + B$

Rewrite as:

2. $F(x,y) = a*x + b*y + c = 0$

Gives: $F(x,y) = dy*x - dx*y + B*dx = 0$

$\Rightarrow a = dy, b = -dx, c = B*dx$

So, if you look back the mid point algorithm is an incremental algorithm and we are working assuming the first octant and we also see today how the other seven octants in the 2D space are handled. And the assumption of the first octant of drawing a line puts a constraint that the slope of the line of the equation y equal to mx plus b as given here m is equal to dy by dx and dx is more than dy in the first octant. That approves the constraint that the slope is less than 1 or the m of the equation is less than 1. And if it is so then of course that given current pixel at any stage of iteration the choice of the next pixel which could be next after current is between any two options east or north east.

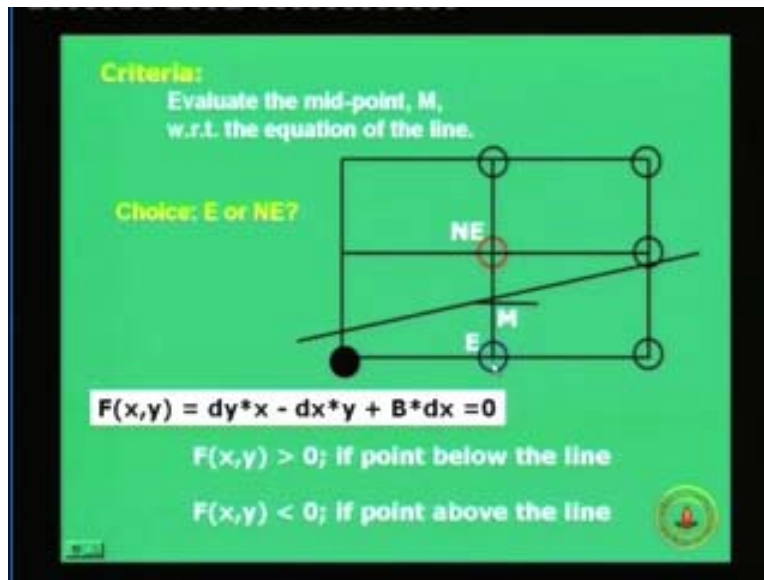
We have also seen a figure in the previous class and we are going to see that. We will revisit it again where the equation y equals mx plus b or dy by dx multiplied by x plus B is the original equation of the line and we try to rewrite the equation in the form $F(x, y)$ is equal to 0 if you try to write where the coefficient small characters abc as you see are the coefficients, the parameters now of the line and we try to rewrite equation number one in terms of equation number 2.

We can basically rewrite it equation number one in this form where the coefficient a is equal to dy , coefficient b is equal to minus dx and the coefficient c is capital B multiplied by dx . So these are the coefficients of the parameters of the line and that means the equation number one takes the equation $F(x, y)$ equals as given here, equal to 0. And we know that what are the coefficients abc now as given in the bottom of the screen. And what is the advantage with the help of this equation in a form as given here, the parametric form $F(x, y)$ is equal to 0 is that, if you select a point which is lying on the line the value of f will be equal to 0 because the value of f or the functional form is derived from the two end points of the line, you must remember that.

And hence if you select a point which is lying exactly on the line which can be done in fact almost only in an analog space with floating point numbers but you are able to select

that integer of floating point if the point lies exactly on the line the value function f is equal to 0. Now the point is above the line or below the line you do have non 0 values of f , one of them will be positive the other will be negative. This is the basic principle of the mid point criteria. And we will see when the value of f becomes positive and when will it become negative depending on the position of a point whether it is above or below the line.

(Refer Slide Time: 8:27)



So the function f splits the line into two parts; the positive half and the negative part below and above the line and the criteria which we used to evaluate, that given this black pixel on the left hand side and at the center of the screen is the current pixel which has been obtained at any stage of iteration of obtaining points for the line from left to right in the first octant. The next two choices which remain would be only the north east pixel as marked in red circle and the east pixel as marked by the blue circle.

So the next choice can only be between any of the two pixels and to find out which is the correct pixel for the next choice what we do is basically evaluate the point m with respect to the line m is the mid point between east and north east so it is mid way between the vertical line between east and north east. And if you note the current pixel coordinates as marked by the black pixel here we know the coordinates of the north east pixel, we know the coordinates of the east pixel as well because they are just integer coordinates shifted in x and y directions by unity and we should be able to get the coordinates of m which could be a floating point number but you do not worry about that for the time being. So the choice as given here will be between east or north east and we have to find out what is this choice. So what we do is take the value of m and substitute in the equation $F(x, y)$ equal to 0.

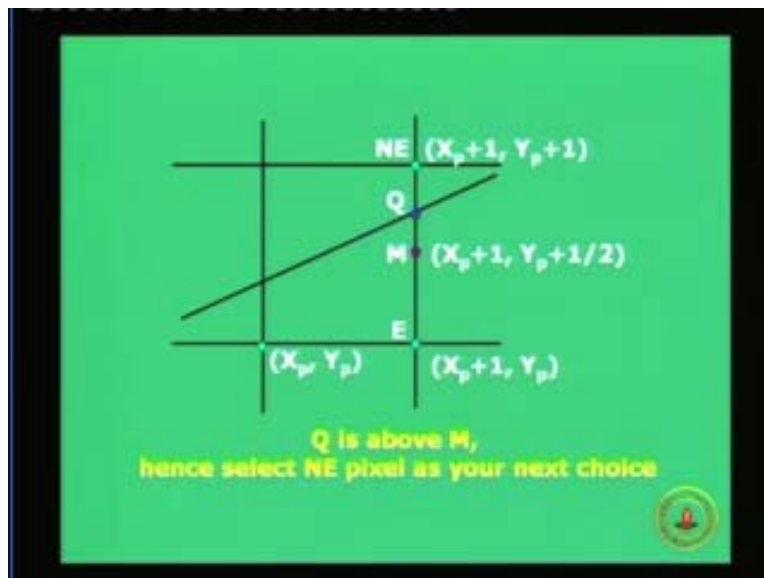
So $F(x, y)$ equal to 0 is given by this expression. And we have to assume here that dy dx and B , these are the three parameters a b and c for the functional form $F(x, y)$ equals to 0

is known to us and after it is known we substitute the coordinates of m in x and y as in this equation and find out whether f is positive or negative.

Therefore, if $F(x, y)$ is greater than 0 we have seen this condition in the last class that if it is positive then the point must be below the line because what will typically happen is, these coefficients will reduce forcing the value of f to go above 0. And if the point m is above the line or the line passes through below m that means between m and east, then of course $F(x, y)$ will be negative. It will be negative because the value of m will have a y coordinate which is more than the value which would have been exactly on the line so this term will now force the entire value of f to be negative.

Just remember that the function f is positive if the point is below the line as shown here and $F(x, y)$ is negative if the point m is above the line. So something which is above the line is the negative world and something which is below the line is the positive world. If you remember that is sufficient, we have now evaluating the mid point to find out whether it is above or below the line, the world above actually is negative above the world below because the entire 2D space you can assume is split by this line into two halves and the world above is actually a negative world on top and positive world below, positive below and negative on the top. That is how the m is lying on the top which is above the line m , we will have negative values if it is below the line it is positive. So positive below, negative above remember that is the criteria which is used for evaluating the line or evaluating the mid point basically, the correct way of saying evaluating mid point m with respect to the line and that helps you to choose to find out easily whether the north east pixel is closer to the line or the pixel east is closer to the line.

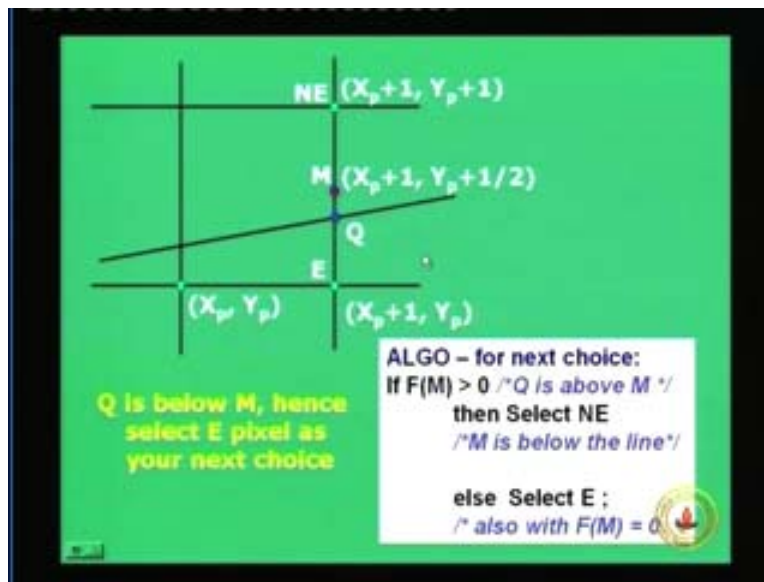
(Refer Slide Time: 9:51)



This is the diagram which shows that the line passes in between of course east and north east pixel and at the current stage of iteration you have chosen in the previous iteration let us say the point X_p, Y_p is the pixel which has already been chosen on the left of your

screen and the line is passing through, it is intersecting the line between east and north east at a point Q. And this point Q is above the mid point on the line. What are the expressions of the coordinates of the point E? If X_p Y_p is the previous stage of pixel then we are trying to find out the current one or X_p Y_p is current and the next pixel must be one out of east and north east. We know that there are two choices that exists. So if X_p Y_p is the previous pixel obtained at the previous stage of the iteration. Then at the current stage what we have is that the east pixel will have the coordinates X_p plus 1, Y_p and the north east pixel will have coordinates X_p plus 1, Y_p plus 1.

(Refer Slide Time: 12:35)



So what should be the coordinates for the mid point m? It will be X_p plus 1 and Y_p plus 1 by 2 that will be the coordinates for the mid point because it is exactly symmetrically located with respect to north east and east pixels. So the difference will only be in the y coordinates, it is not in the x coordinates any more but the line L which is the line of interest to be drawn, it is now intersecting the vertical axis between north east and east at the point Q. And we assume now as shown in this figure that if Q is above M the world above as we have seen is positive and so what we see is, you will have a positive value and you select north east pixel as your next choice. The reverse will happen if the point Q is below M.

Remember, in the previous figure we had, in the previous slide Q was above M, now Q is below M or the mid point is above the line and the Q is either below M or the mid point is above the line. We have to select east pixel as our next choice because what will happen is basically the value of f will be come negative now and the Q is closer to M than in the north east pixel. In the previous figure Q was closer to the north east pixel so obviously you have to choose north east in this case. The Q is closer to E than north east so you choose east. But how it is closer and which one is closer to Q north east or east is basically decided by the value of f after substituting the value of the coordinates of M in the functional form of the line $F(x, y)$ equal to 0. So we can write the mid point criteria in

the form of an algorithm now and the algorithm for the next choice, this is not the full algorithm for the mid point, it is just the criteria for choice which is in fact the kernel or the key part of the algorithm or the code which you have to write is the algorithm for the next choice of the pixel is, if f of M that is the functional form or value for the line evaluated at the mid point is greater than 0 that means Q is above M . That is the figure which you have shown in the previous slide, not this particular figure but we assume that in this case the functional value $F(x, y)$ equal to 0 substituted by the mid point coordinates M is positive or greater than 0 then we know that the Q is above M or M is below Q .

In whatever way you look at it Q is above M and closer to the North east pixel and M is below Q and it is under this condition where you will have f of m is greater than 0 and then you select north east if it is positive and if it is negative else you select east that is this particular figure where Q is below M or M is above the line Q . So this is the mid point criteria written in terms of the algorithm which is the key to choose your next choice. And once you are through with the next choice you apply the same for the next iteration. After you have selected either east or north east, one of these two become your current pixel of iteration and then you move to the next iteration where the value of X_p to X_p plus 1 and X_p plus 2 is your x coordinate and you then find out with respect to the choice what is going to be your next to next, east or north east pixel.

So this is the mid point criteria, the basic principle behind which the Bresenham's algorithm or mid point algorithm works. And now we will go through few sorts of equations to find out where is the integer arithmetic coming in and which speeds a power method to obtain the sequence of points. We move to the next slide where our basic criteria is to evaluate the mid point M using a decision variable because that is what you evaluate. You need to find out whether M is below or above the line using criteria and based on that whether the functional value f is positive or negative, you choose east or north east. So just remember this principle and we define a decision variable that is key to the algorithm, we define a decision variable d which is equal to $F(x, y)$ and the decision variable d when evaluated at the next mid point M .

(Refer Slide Time: 16:24)

Evaluate mid-point M using a decision variable $d = F(X, Y)$;

$$d = F(X_p+1, Y_p+1/2) = a(X_p+1) + b(Y_p+1/2) + c;$$

at M ,

Set $d_{old} = d$;

Based on the sign of d , you choose E or NE.

Case I. Chosen E:

$$d_{new} = F(X_p+2, Y_p+1/2)$$
$$= a(X_p+2) + b(Y_p+1/2) + c$$
$$(\Delta d)_E = d_{new} - d_{old} = a \quad /* = dy */$$

That means assuming that at correct stage of iteration or the previous stage of iteration X_p Y_p is the current coordinates, the next mid point will have coordinates as given here within the bracket of $F(X_p$ plus 1) that means x shifted by 1 and Y_p plus $1/2$. Those are the coordinates for the value M and if you substitute the value of M in this and I hope you have written and you are remembering the expression of f which has been written earlier and substitute that and this is the expression which you have. This is the expression which you have for the decision variable d evaluated at point M .

And let us say you have set this d old at equal to d . This is the old variable defined new, old variable d old equal to d and then based on the sign of d you actually have do not have to look into the value of d anymore, you have to find out whether the decision variable d is positive or negative at M and based on that you choose one of the east or north east pixels. So assuming you have chosen one out of east or north east pixels based on the sign of d you will have two different conditions or two different cases.

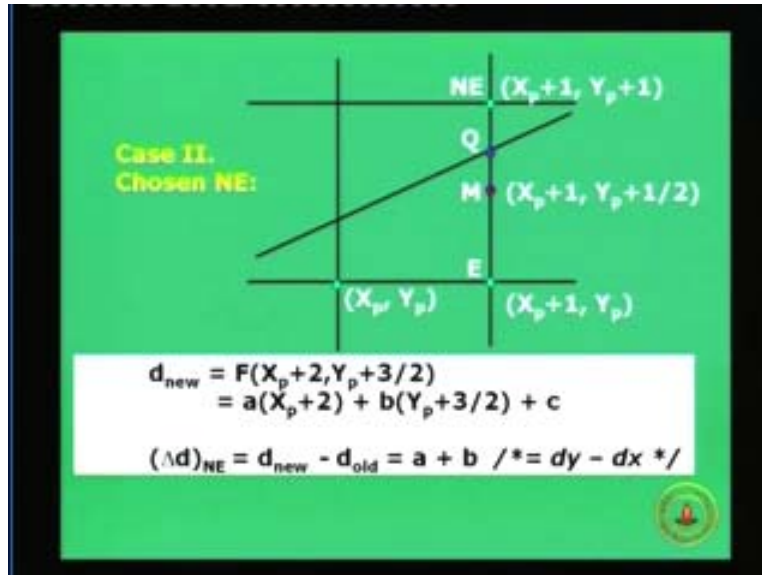
We look at the case one where you have chosen east as your next choice of pixel that means when you are choosing E you must remember that the line is passing below the mid point M or the M is above the line basically the decision variable is negative. If you remember those two criteria, the sign, the world above is negative and below is positive. So, the line is passing below the mid point M and it is closer to the east pixel and that is the reason why you have chosen E .

We will look into the other case where you have chosen north east. So look at the first case, let us look at the equations, once you have chosen east the d new that means the d new is the decision variable evaluated at the next to next mid point. After the current mid point is over we are looking into the next decision variable d new where the mid point will now have coordinates. It was X_p plus 1 Y_p plus $1/2$, so what will happen is, after you have chosen east, you remember the grid situation, the figure given to you earlier, it will

come back again, it will be X_p plus 2, $(Y_p$ plus 1 by 2), why? Because $(X_p$ plus 1) plus 1 so you will get $(X_p$ plus 2) and the y coordinates of the previous mid point remain the same. So it will be Y_p plus 1 by 2. So break open the expression and rewrite it, you will get this. And the expression is given as $a(X_p$ plus 2) plus $b(Y_p$ plus 1 by 2) plus c that is the new expression d new. That is the value of d evaluated at the next to next mid point after you have chosen east. And we will say the difference is Δd under the subscript E indicates that it is under the case one where you have chosen east, that is, the difference in the d will be d new minus d old where d new is given by the expression as given here and d old is given at the expression, because you have set the d to the d old earlier, so it is given here on the top right of the screen. This is the expression of d old and this is the expression here which is d new. So subtract this from that you will get a . That is all because the rest of the terms cancel out, c cancels with t $b(Y_p$ plus 1 by 2) cancels with $b(Y_p$ plus 1 by 2) here, you will be left with only a , because a times X_p also gets canceled. So the subtraction of d new minus d old gives you a simple term a . And you remember a is equal to dy as given to you earlier and that is an integer quantity. So the first ordered difference Δd E because that is a differential term between the new value of d and the old value of d is given by a .

So Δd E subscript chosen, first case chosen is E is equal to a . So let us evaluate the d under the second case where you have chosen north east. When do you choose north east, line is above the mid point or the mid point is below the line and when the mid point is below the line the functional value is positive. So the functional value is positive or the decision value is positive and that will definitely give you the condition that the sign of d is positive and the line is closer to north east. So you have chosen the north east pixel. And if you have chosen the north east pixel let us see what happens to the same set of equations. So we look at case two where you have chosen north east and the figure is given here, so the mid point M is below the line or the point Q is the above the mid point which is closer to north east and remember the expression of d old as given in the previous slide.

(Refer Slide Time: 19:42)



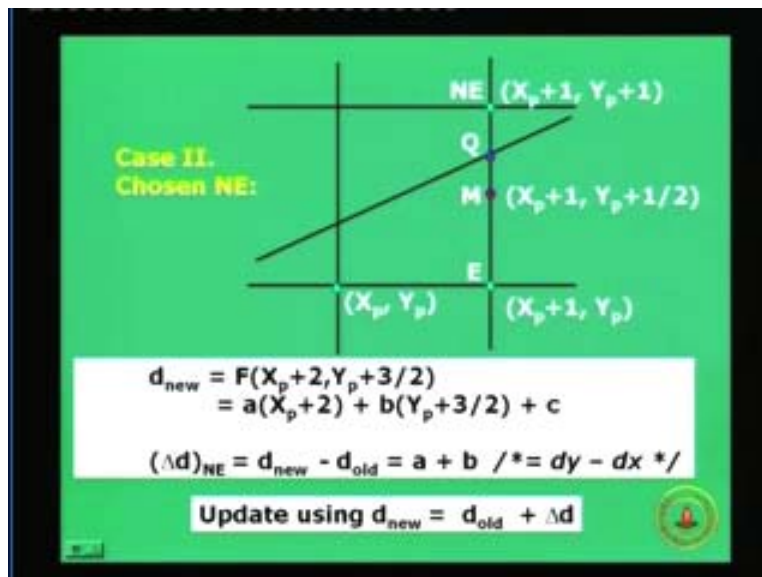
The new mid point for d new will be, if this is X_p plus 1 the next mid point will be at X_p plus 2 and Y_p plus 1 plus 1 by 2 which will be giving you Y_p plus 3 by 2. That will be in the mid point. If this is X_p Y_p this is X_p plus 1 Y_p plus 1 by 2 for the first mid point. The next iteration mid point will be moving X by 1 coordinate and also y by 1 coordinate. So it will be X_p plus 2, 1 plus 1 and 1 plus 1 by 2 will give you 3 by 2 or 1.5, so that is the new mid point under the second case where you have chosen north east. You substitute these two x and y values onto the value of f and this is the expression which you get for the d new. This a multiplied by X_p plus 2 plus b multiplied by (Y_p plus 1.5 plus c). So this is the new expression of d new and I hope you have written the expression of d old, so we subtract d old from d new, subtract these two expressions and get the expression of the delta d under north east, under the case that you have chosen north east and that expression difference will give you a plus b. And you know what are the parameters a and b? They are nothing but in common [{21:08 }] it is nothing but dy and dx.

So as you see from the previous slide here delta d under case one where you chosen east, the delta d is a and under case two where you have chosen north east, the delta d north east is dy minus dx or a plus b. And both dy and dx are nothing but what? They are the difference in the, the dy and dx are the difference in the x and y coordinates of the starting point and finishing point of the line. And since the starting point and finishing point of the line, the coordinates are integers, their differences will also be in integer coordinates and hence these differential delta d terms of the decision variable which is in fact where you update the decision variable d also will be integer coordinates. So if you need to update something incrementally and if that increment arithmetic can be made using integer values, I did mention you earlier that the algorithm will be very fast, the program will run at a very top speed and the equations will tell you that the incremental variable of d.

We will see how that is used now within the algorithm and that is based on the integer values and it will help you to really speed of the algorithm. But just remember now that based on the sign of the decision variable d to choose east and north east and based on whether you have chosen east in case one or north east in case two the delta d_e will be equal to dy or delta d north east will be dy minus dx . You remember these two expressions and you remember the mid point criteria for choosing the east or north east pixels based on the sign of d is sufficient for you to write an algorithm or code yourself for drawing the lines. And if you remember these two you will be following the algorithm which I will show you next.

Remember two things, mid point criteria that is based on the sign of d you choose whether north east or east pixel and once you have chosen east or north east pixel, the differential term which you have to add to the decision variable d will also be having two values, in one case it will be dy and in another it will be dy minus dx as given in the two expressions we have seen in the previous two slides. Just remember the mid point criteria and expressions for the delta d increment and these two different conditions of choice, east or north east is sufficient for you to follow the algorithm which we will cover next.

(Refer Slide Time: 23:40)



So in each case the last line will say that we need to update the decision variable d to derive the new value of function d new from d old using the incremental value of delta d , delta d have two different expressions; one for the choice east and another for the choice north east and that is what we have seen. So the mid point criteria: We just revisit in the form of the algorithm now, the decision variable d is of course I will give you an algorithm form later on but we visit entire criteria and the choice and the delta d now, d is F of M which is given by this expression evaluated at the mid point X_p plus 1 Y_p plus 1 by 2 if d is positive choose north east else if d is less than equal to 0 choose east. Remember, we are talking of a positive or negative and we have to worry what happens if in a situation which may occur very very really, in very real situation you will have an

integer pixel exactly lying on the line. But if it exactly lies on the line you have to take a decision.

In fact the decision says that if the mid point is lying on the line, your next choice of north east or east pixel are equidistant from the line. M is now exactly on the line and it is exactly mid way between east and north east. If these are the north east and east pixels let us say and the line passes exactly to the mid point of the line and this intersection point Q is exactly mid way between north east and east, then you can choose any one of them. You can choose any one of them is what the algorithm says but in terms of implementation one must say here that you must be choosing this east or north east.

You can choose any one of them but you must uniformly do that through your implementation and algorithm. Otherwise you will keep drawing different lines for different situations. So do not keep choosing randomly when d is equal to 0, in one case you choose north east, another case you choose east now. Your graphical software which you designed must either choose consistently and uniformly north east or east in this algorithm which we have seen. We say that if d is equal to 0 choose east.

So throughout we will follow that if d is negative or equal to 0 we choose east. If d is positive we choose north east. And in brief again we see in the case when you have chosen east what you have to do is the following steps; increment the M by 1 along X direction that the x coordinates of the mid point is incremented by 1 which gives us the new value of d new and that is obtained by incrementing the delta old by a value dy.

(Refer Slide Time: 26:26)

Midpoint criteria

$$d = F(M) = F(X_p+1, Y_p+1/2);$$

If $d > 0$ choose NE
else /* if $d \leq 0$ */ choose E ;

Case EAST :

increment M by 1 in x
 $d_{new} = F(M_{new}) = F(X_p+2, Y_p+1/2)$
 $(\Delta d)_E = d_{new} - d_{old} = a = dy$
 $(\Delta d)_E = dy$

Case NORTH-EAST:

increment M by 1 in both x and y
 $d_{new} = F(M_{new}) = F(X_p+2, Y_p+3/2)$
 $(\Delta d)_{NE} = d_{new} - d_{old} = a + b = dy - dx$
 $(\Delta d)_{NE} = dy - dx$

We know that delta d east is d new minus d old which we have derived already is equal to a is equal to dy and delta d east is dy and in the case of north east the delta d north east is given by dy minus dx. That is the one difference in last two lines delta d under the case of east is dy delta d or the difference of the decision variable under the case that when you

have chosen north east is dy minus dx and that is the main difference. The other difference is, you have to increment M by 1 in both x and y direction in the case of north east. In the case of east you increment x coordinates only by 1. So these are the two main differences in the choices and what you start to the value of d ?

(Refer Slide Time: 27:00)

What is d_{start} ?

$$d_{start} = F(x_0+1, y_0+1/2)$$

$$= ax_0 + a + by_0 + b/2 + c$$

$$= F(x_0, y_0) + a + b/2$$

$$= dy - dx/2$$

What is the d at the initial position? Because you know the starting point which is the starting point of the line given by x_0 y_0 and you keep incrementing by, if you keep choosing east and north east pixel. You have a starting point, if you have a starting point in some corner and this is the finishing point let us say, so you have keep traversing by choosing east or north east pixels and find the addressable pixel closer to the line, this is the line let us say and then reach the panel destination x_1 y_1 which is the ending point of the line but you have to start. This starting point is noted, what about the next? So evaluate d at the starting point, you evaluate d at the starting point which we say is the d start.

So the d start, if we look into the expression is given as f of x_0 plus 1 y_0 plus 1 by 2, why? It is because you have to evaluate the d at the start for the first mid point. First mid point will be one more in the x coordinate and half vertically along the y coordinates. I shall substitute these expressions and then try to re-substitute back the value of f x_0 y_0 . You have this expression as ax_0 plus a by by_0 plus b by 2 plus c .

I hope you remember the expression of f , substitute these two, this is the expression you have and you just borrow certain terms and find out that this can be easily written as fx_0y_0 plus a plus b by 2. It can be rewritten in this form and you know f of x_0 plus y_0 what it is? It is the starting point of the line. And the starting point of the line is lying on the line. So of that is so then what you may have is you should have the value of fx_0y_0 which is the starting point and a point lying on the line, the value of f is equal to 0. So, this term cancels out, the value of this term is equal to 0 and you will be left only with $(a$

plus b by 2). And a and b are nothing but we know that they are dy and minus dx. So the value of the d at start is given by dy minus dx by 2. That is the value of d at the start and then you keep looking at the sign of the d.

We have seen the mid point criteria based on sign positive, negative or 0 you choose east or north east, keep incrementing d using delta variables, keep on doing that iterations till you reach the end point, so that is the basic idea of entire algorithm. A choice of mid point criteria, incrementing the decision variable using integer arithmetic and then keep on doing this incrementally at each iteration. But if you look back into the slide which I have just shown now the expression of d start is given by dy minus dx by 2, dy and dx which also comes in the expression of delta d are integer coordinates but dx by 2 is not guaranteed to be an integer, is it not? You divide an integer by 2 you have not guaranteed to have an integer value.

If dx by 2 is a floating point number; in fact you are going to start d as a floating point number itself. So we have a problem that d start itself should be an integer value, if it is not, it is a floating point value then the entire algorithm turns to a floating point operation and that will make the algorithm very slow. So we have to start making d start of the starting value of the decision variable d as an integer itself. You need to get rid of this fraction and see what we end up with all the variables. How to get rid of this fraction? Well a simple suggestion is, multiply the value of f by 2. If you multiply the value of f by 2 if f of x_0y_0 is equal to 0 then 2 of x_0y_0 is also equal to 0. So the d will be 2D instead of d itself and you will get the value of d as 2 dy minus dx.

(Refer Slide Time: 30:32)

What is d_{start} ?

$$d_{start} = F(x_0+1, y_0+ 1/2)$$

$$= ax_0 + a + by_0 + b/2 + c$$

$$= F(x_0, y_0) + a + b/2$$

$$= dy - dx/2$$

Let's get rid of the fraction and see what we end up with for all the variables:

$$d_{start} = 2dy - dx ;$$

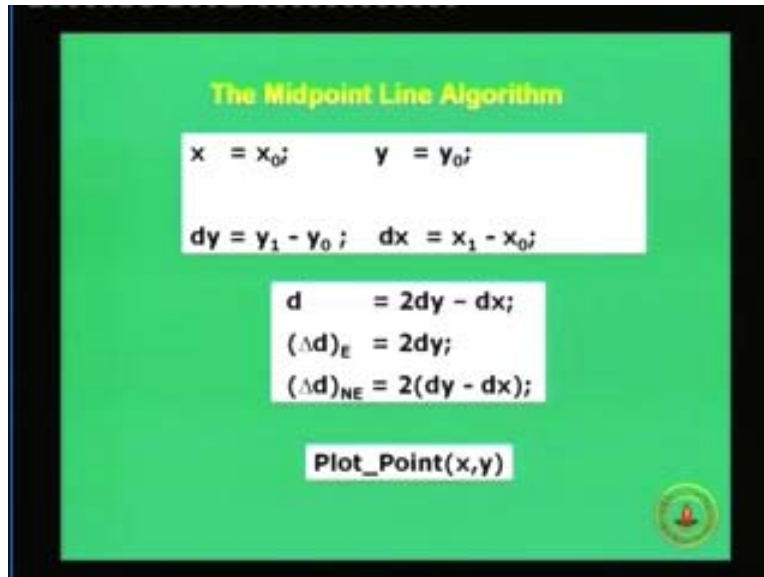
$$(\Delta d)_E = 2dy ;$$

$$(\Delta d)_{NE} = 2(dy - dx) ;$$

You multiply all of the variables, the starting and the increments etc, everything by 2 and now you see that the decision variable at the start and the increments itself which were again integers multiplied by 2 will give you integer. So you have integer arithmetic now starting from the initial value or starting value of d which is called the d start or d in it.

You also called it d in it in the initial value of d. So these are the expressions which you need to remember which have to be used in the algorithm. Let us look into the mid point line algorithm now.

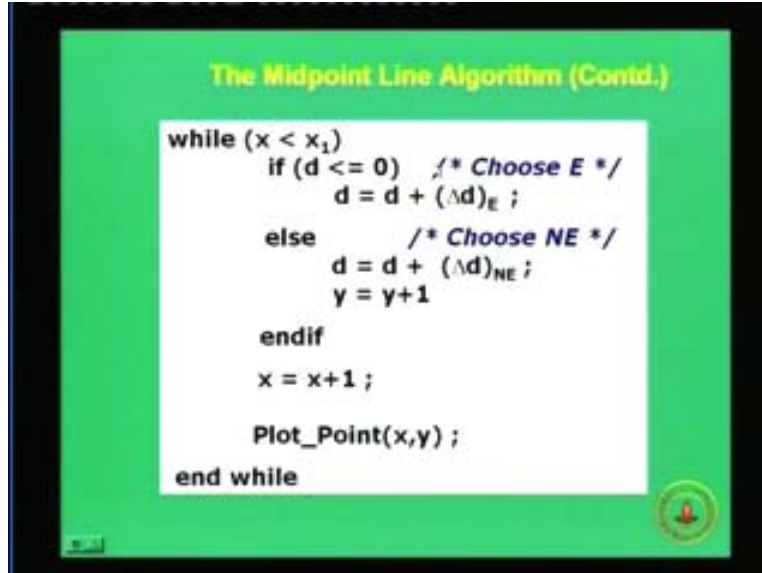
(Refer Slide Time: 32:18)



We will look at the initial conditions first of the algorithm as given by the equations and that is given by the x, we start with x equal to x_0 , Y equal to y_0 which is the initial condition that $x_0 y_0$ is the starting point. So you initialize the current coordinates to be equal to $x_0 y_0$. And we also have the decision variables which have to be defined. Before that you define dy which is difference between y coordinates starting point and ending point of the line y_1 minus y_0 , dx is the x coordinate from y_1 minus y_0 , dx is the x coordinates x_1 minus x_0 . So that is the first part of the initialization, then the initializations continue where the same set of equations are repeated. W

e have the d start equal to $2dy$ minus dx , that is the d start, the initial value of d you do not have to write the d start here, the d starts with the value of $2dy$ minus dx . And the two incremental values which are also integer coordinates delta d of E will be equal to $2dy$, del d or under the case of north east will be $2dy$ minus dx . So that completes the process of initialization and we can start with the loops. Of course you first start plotting the starting point itself which is a plot point command and we will see later on that this plot point commands not only plots the point at the first octant, it also handles the other cases if the line is not in the first octant. We will see that.

(Refer Slide Time: 32:14)



So we are only worried about the first octant now, but we will also see how the other seven octants are handled. So the first point is drawn because at the initial case x is assigned to the x_0 , y is equal to y_0 and it has not been changed so far under initialization and so the first point which will be drawn by these command plot point, assume that low level graphics primitive command given to draw a point with a certain default color or shade with the value of x , y will be having the value x_0y_0 . So the first point is drawn which is the starting point. Then, this is the rest of the part of the mid point line algorithm.

And the entire part of the algorithm is given here and the rest of the part is given after the initialization. You can actually put everything in a while loop, you can also put everything in a fall loop, it does not matter. And you keep on doing these iterations till x is reaching x_1 . We stop when x reaches x_1 and so as long as x is less than the x coordinate is been incremented by unity from x_0 towards x_1 and as long as x is less than x_1 we keep on repeating this while loop. And what do you do inside the while loop as long as x is less than x_1 ? For that we look into the sign of d if it is negative or equal to 0, in comments I have put choose east, you choose east and then what you do basically is increment d by Δd_E which has been already initialized and the particular expression as two times dy . It is already initialize here, so increment d by that particular incremental variable which is again integer. So d is an integer Δd also an integer that is how this is. Else means you choose north east that means d is positive, you choose north east in this condition because the line is above the mid point M or basically the point M is below the line and hence you have got a positive value of d . In that case basically you choose north east means you first increment the decision variable d here and also increment y because for both cases you need to increment x which is also put under the expression here that you basically increment x outside the if condition.

In one case only you increment y when you are choosing the north east and then plot the point again. So this xy , if you have chosen east it will be only incremented in the x direction by 1, if you have to choose north east based on the sign of d then at the else condition you enter what will happen is you have incremented y here then you will again increment x and you have basically chosen the north east as the current point of iteration of the pixel to be drawn or addressed and you use the plot point with xy . So you keep repeating this loop, keep on checking the design of the decision variable, based on that you choose east or north east, increment the decision variable d , increment x in both cases because you are always going from left to right in integer coordinates increasing by unity. Only when you choose north east your incrementing by 1 as you can see here, you do not use these y is equal to y plus 1 under east because you are only incrementing along the x direction, x coordinates will increase, y will not if you are choosing east. If you choose north east y is incremented here and the x will be again incremented in both the cases.

So this loop keeps on incrementing till x reaches x_1 . You do not do this as a loop any more when you are reaching x equal to x_1 but as long as x is less than x_1 you look into the condition put under the while loop, you have to keep on incrementing this decision variable and x and y coordinates, y depending upon whether you have chosen east, only in the case of north east you increment y and the choice of the mid point criteria is only looked into the sign of d . So the three conditions were; look into the sign of d and then choose east or north east and increment d . And then of course the corresponding x and y coordinates obtain.

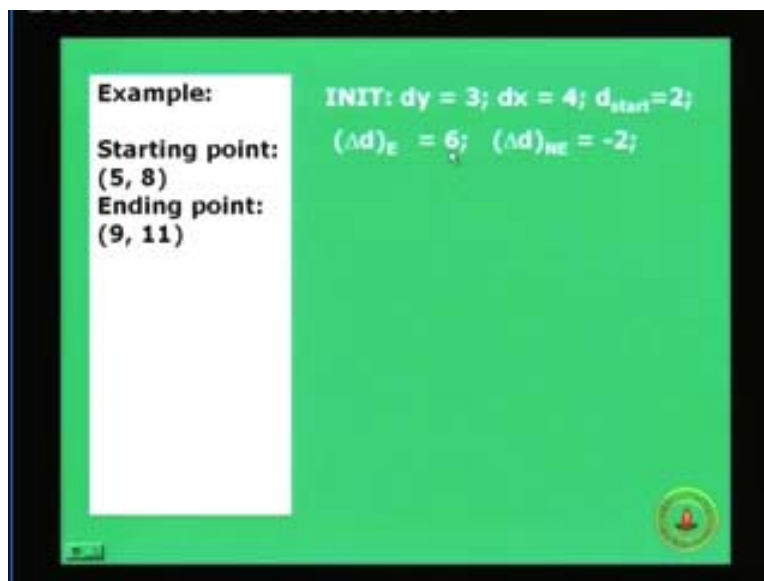
This was the basic principle behind mid point criteria or Bresenham's algorithm for drawing a line and they have seen the algorithm starting from initialization of the d x and y through the while loop there is only incremental integer arithmetic of addition. There is no round function and there is no floating point. And this while loop will run very fast if you have dedicated hardware in the system and be functions for integer arithmetic only incrementing by integers or incrementing xy coordinates, one of them or both by one. And this will zip through and generate a few hundred points if necessary for a line given from the bottom left of your screen to the top right.

Let us say of your screen you can actually draw a set of hundred points in a very fast time, it should be definitely a fraction of a second not more than that. And that is possible because the use of Bresenham's mid point criteria of evaluating the mid point based on the sign of the decision variable which is again integer and incrementing using integer, incrementing xy coordinates by integer. So it is integer arithmetic based and that is the thing which we would exploit. Why we could exploit that? We could exploit that because we were worried in the last lecture in the beginning of drawing lines that we are already with floating point numbers and that it will slow the system, we had to use round functions and all that. Those are all the reasons why the algorithm was slowing down. We have thrown all of that now and pure integer arithmetic and that you are able to use that because the entire space is discretized, digitized, it is a discrete space with integer xy coordinates.

Since we are only moved with integers why worry with a floating point number although the value of the slope of the line M could be a floating point number we do not worry about that, start with the integer and keep incrementing with integer values with decision variable d , look into the sign of d and find out the choice east or north east and then side what is your next point. So that is the basic idea and we have seen the algorithm. You please rework out the equations yourself to find out the decision variables d under east or north east and also d start and how we multiplied by 2 to get rid of the floating point at the initialization face itself. Mid point algorithm is assumed to be understood but there are a couple of points here which one must mention. The first point which we must notice or take a fact is that we assume that the points are only in the first talk time.

Now, what about the rest of the seven octants in the 2D space? That has to be handled and that is number one. And of course could there could be other points when the slope of the line is very low that means close to 0 or very high in fact it is very close to 0 it is called a shallow line. We also have some problems and some special cases and we also look at the problems of aliasing of a line which is not considered in Bresenham's algorithm. Aliasing of a line takes place due to the digitized environment, what the Bresenham's line algorithm solves or gives you a solution in a very fast time is used find out the set of addressable pixels on the line which are the most closest to the line, which should be drawn or illuminated to give you the effect of the line when drawn on the screen. So we will take a small example of the Bresenham's algorithm and then move to see how the rest of the octants are addressed.

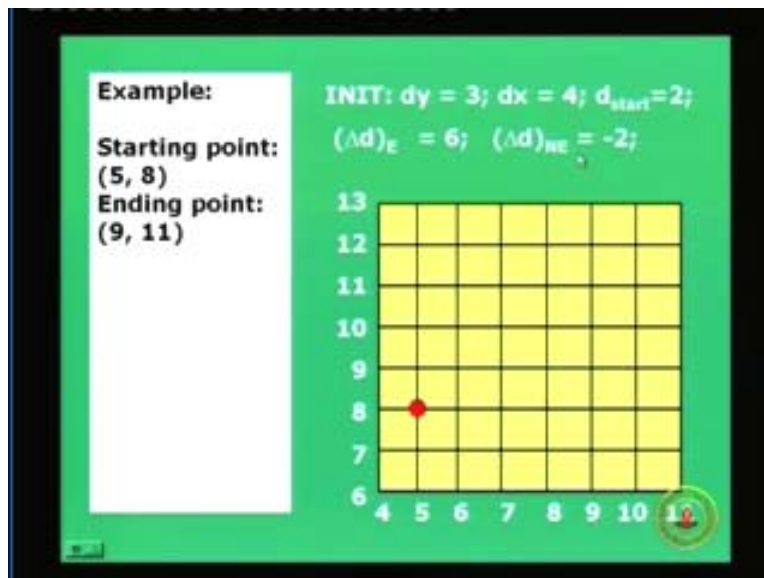
(Refer Slide Time: 41:02)



Let us say, you can take an example and apply Bresenham's algorithm to this and say the starting point is 5, 8 and the ending point is 9, 11 of the line. So $x_0 y_0$ is 5, 8, x_0 is 5, y_0 is 8 and x_1 is 9 and y_1 is 11. If that is so, the initializing conditions are dy which is y_1 minus y_0 , 11 minus 8 so dy is equal to 3. What is dx ? It is x_1 minus x_0 , 9 minus 5 will be equal to 4. So dx is equal to 4, dy is equal to 3. What is d start? Do you remember the

expression of the d start or the initial condition of d ? It is $2dy$ minus dx , 2 multiplied by dy minus dx and that expression will give you 2 multiplied by 3 minus 4, so it is 6 minus 4 d start is equal to 2. You see all integer values here you need to have the initialized conditions in the case where you have the expressions of Δd_E and Δd_{NE} . What is the expression for Δd_E ? It is $2dy$ if you remember that. So $2dy$ is 3 multiplied by 2 which is 6. And what is the value of Δd_{NE} ? Do you remember that? Let us look back and see the expressions Δd_{NE} is 2 multiplied by dy minus dx . You see the expression of Δd_E is $2dy$. We have already seen that the value 2 multiplied by 3 is 6 and this is 2 into dy minus dx . Let us go back to the example and we will see 2 into dy with 2 into 3 minus 4, 3 minus 4 is minus 1. So you get a minus 2 for Δd_{NE} . So the entire initialization is over. The initialization is over and we start iterating now.

(Refer Slide Time: 42:36)



So the first point is the plot point x, y which is you start with 5, 8 so that is drawn. That is the red pixel at x coordinate 5 and y coordinate 8 is purely drawn on the left hand side of this square grid, some small portion of the screen where x starts from 4 to 11, y from 6 to 13, a small part of the screen is shown because you start from 5, 8 and let us hope you finish at 9, 11. If you do not finish at 9, 11 there is something wrong in the algorithm or we have made some mistake in the calculation so far. We will see with a little bit of suspense whether you really wind up with the integer coordinates 9, 11 and this is the finishing point.

What is the initial value of d ? d is equal to 2. Successive steps involve d equal to 2 which is a positive number. Positive number means the mid point is below the line. So mid point is below the line or you do not even worry about that just look into the algorithm and you know if it is positive you have to choose north east because north east is closest to the line. So if you have chosen north east 5, 8 is the first starting point or the current point of the iteration at the start, the next point will be 6, 9. There are two choices, you can either

choose 6, 8 or 6, 9 and north east is 6, 9. Once you have chosen that 6, 9 and you have chosen north east you have to increment d by this value Δd north east. So what will be the value of d ? It will be d start equal to 2 minus 2 so d will have a value 0, d is equal to 0 will actually give you the next choice as the east pixel. You know if the value of d is 0 or less than 0 then of course it typically means that the line is passing closer to east, the mid point is above the line and so the next choice is 7, 9. So that is marked in red, I hope you can see that 7, 9 easily. Thus, two more iterations are over after the initial point 5, 8. So 7, 9 is east, so you have to increment d by Δd E which is the value 6 here, so 0 plus 6 will give you 6 so that is the new value of the decision variable d which is given as 6. And it is positive which by the decision mid point criteria tells you that you have to choose the north east point which is with coordinates 8, 10. That is the point which is chosen now under the third iteration.

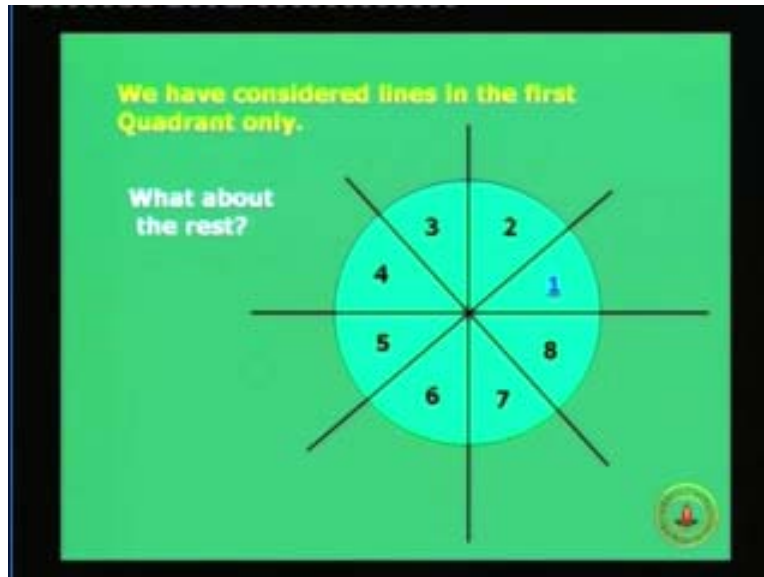
The first iteration gave us 6, 9, second 7, 9 and next 8, 10. We are almost reaching the end point now. So now 8, 10 is north east, so Δd north east is minus 2. You have to add this Δd north east incremental variable to the decision variable d 6 minus 2 which will give you d is equal to 4. The d is equal to 4 is what you get and there again d is positive and that gives you the value 9, 11. So that is the point which you draw as the finishing point of the line and here you find out that you have reached a value x equal to x_1 where x is 9 which is equal to the ending point x coordinates 9. You do not do any more iterations, so our suspense is over. We have started with 5, 8 and we have reached 9, 11 without worrying about the ending point.

So that means the algorithm is correct. We have not made any mistakes in the calculation. This is just a sequence of steps taken by example which you should workout yourself and see that you start with the decision variable, take any two points 5, 8 and 9, 11 is taken in this case. But you can take any two integer coordinates, only assume you are in the first octant slope is less than 1 and that is what you should assume and then just apply little bit of steps for the decision variable d based on the sign and the incremental algorithm of Δd east or north east and keep incrementing and select your east or north east successively and see that if you are reaching the end point. I will repeat the set of successive steps once again.

This was the starting point and then first variable d is equal to 2 so choose 6, 9 then d is equal to 0 you choose 7, 9 then you have d is equal to 6 gives you 8, 10 and then again d is equal to 4 you actually choose the last point 9, 11. So that is the example which you have taken, I do encourage that you should take some more examples to get used to applying this algorithm. And this was the line, the blue line dash shows the line on the screen which is the ideal line but I will say in the analog world non-discretized space was the continuous line shown in by the dashed blue line and you can see that the points are spread around. And the three points which you have chosen between the starting and the ending point of the line are the closest possible addressable pixels, addressable pixels are the intersection points of the grids in this case and these three points are the once which you are close to. You will not be able to find any other point which is closer than these three set of points to draw a line. So that is an evident proof that the Bresenham's mid point criteria works really very well.

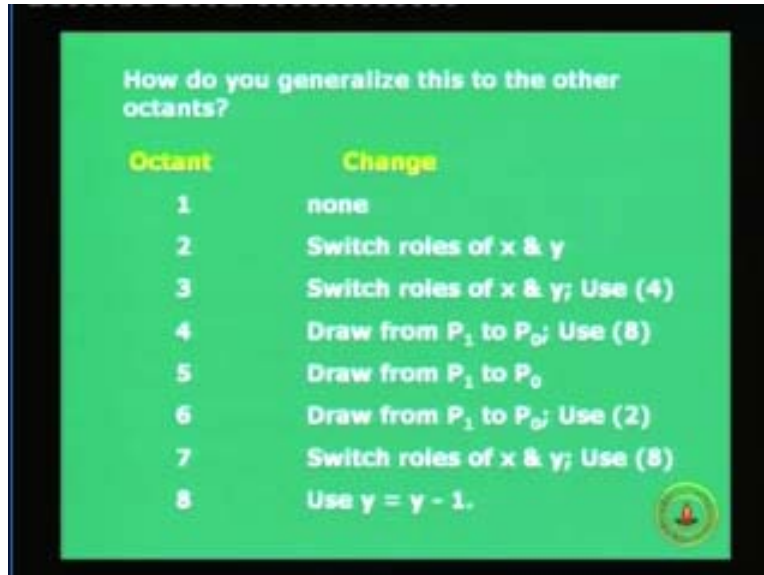
Now let us think about the other octants. In the time left in this lecture we will worry about seven other octants because we have only worked on the first octant slope which is less than 1 and will worry about the seven other octants.

(Refer Slide Time: 46:58)



So we look back into the figure, this figure if you remember that we had shown this figure earlier in the last class where we have considered lines only in the first octant marked in blue color as 1 and what about the rest. That means if the lines lie in the any of the other seven octants starting from 2 to 8 that means the slope could be more than 1; it could be negative and so on. The starting point could be on the right hand side with respect to the left hand side, that means the x_0 is more than x_1 what do you do? Well you have to make some initialization as an overhead to change the scenario to two quadrants and that is number one. And then apply the Bresenham's algorithm, because if all the other seven situations can be converted using some sort of a symmetrical nature of this distribution of octants then and every line can be post equivalent to or mapped on to quadrant number one and you get the integer coordinates in quadrant number one and then map it back and put it in the other quadrant as the case may be depending upon whether you have 2 3 4 5 6 or 7 or 8 quadrant then you have solve the problem.

(Refer Slide Time: 47:59)

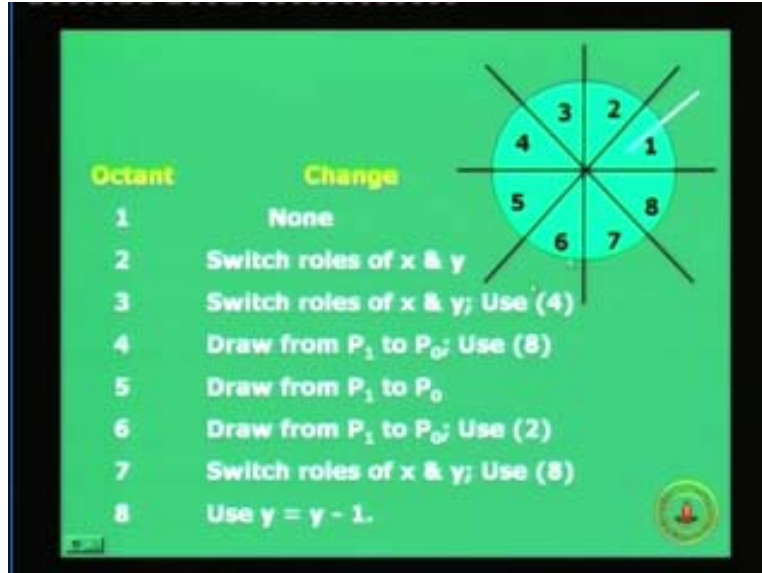


How do you generalize this to the other octants?

Octant	Change
1	none
2	Switch roles of x & y
3	Switch roles of x & y; Use (4)
4	Draw from P_1 to P_0 ; Use (8)
5	Draw from P_1 to P_0
6	Draw from P_1 to P_0 ; Use (2)
7	Switch roles of x & y; Use (8)
8	Use $y = y - 1$.

What you do? How you generalize this to the other octants? These are the cases. So this must be as an overhead where you have to write a small code before you actually apply the Bresenham's algorithm and what you do basically is take your line into the first octant and get the points and then apply it back. So when you are in the first octant do not make any change because you already know the Bresenham's algorithm here. In all the other octants which we see here there are basically two important statements; one is switch roles of x and y is the statement which you find for octant number 2, 3 and 7 and this is one statement. The other statement which you find is draw from P_1 to P_0 . I will try to discuss a few of these cases in the available time and I leave it as exercise for you to check for the other octants as well and see how this helps you. Let us move into this figure, you remember how you solved the problem in octant number one where the line is drawn.

(Refer Slide Time: 48:57)



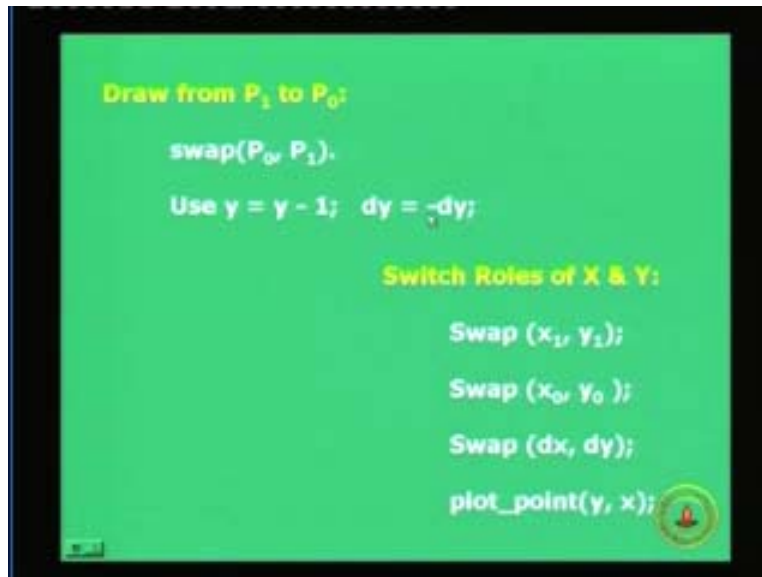
What about octant number 2? Octant number 2 tells you that the line is having the slope more than 1. If the line is having slope more than 1 what you can do is, the change which you need to put is switch roles of x and y. That means the x coordinates should be considered as y coordinates now and vice versa. Make this change; what you're basically doing is mapping the quadrant two on to the quadrant number one. Or basically reducing the slope from more than 1 to less than 1, find the coordinates in the first octant but seems x is now y and vice versa you will be actually plotting the coordinates in the second quadrant origin itself. So this second quadrant is easy to visualize, when you switch roles x and y you are basically done with the job.

What about quadrant number 3? This is a unique case where the slope is not only more than 1 but it is negative. It is negative and then you say that you switch roles from 1 by 2 x and y, not only do that but use number 4 which says draw from P_1 to P_0 . Now quadrant number 3 forces you to a scenario where you are not drawing from left to right anymore.

We have discussed about an initial point to your left and the final point to your right, now it has been swapped, swap means this is the initial point and this is the final point and you are drawing in the third quadrant mind you and the slope is also negative and you are drawing from the starting point to the finishing point and this is the starting point which is to your right and the finishing point is to your left. So you have to reverse the role, you have to swap the roles of not only x and y to bring the slope less than 1 but you also have to actually draw from the reverse, draw from instead of having P_0 to P_1 you draw from P_1 to P_0 , apply the reverse roles basically drawing left to right and then of course changing that. That is what is done for octant number 3 where you not only switch roles of x and y use number 4 which says draw from P_1 to P_0 and then use number 8 where it says keep decrementing y instead of incrementing y. We have incrementing x and incrementing y, in this case you increment x and decrement y. So I leave the rest of the octants for you to check up, we have only discussed quadrant number 2 and 3. I leave the

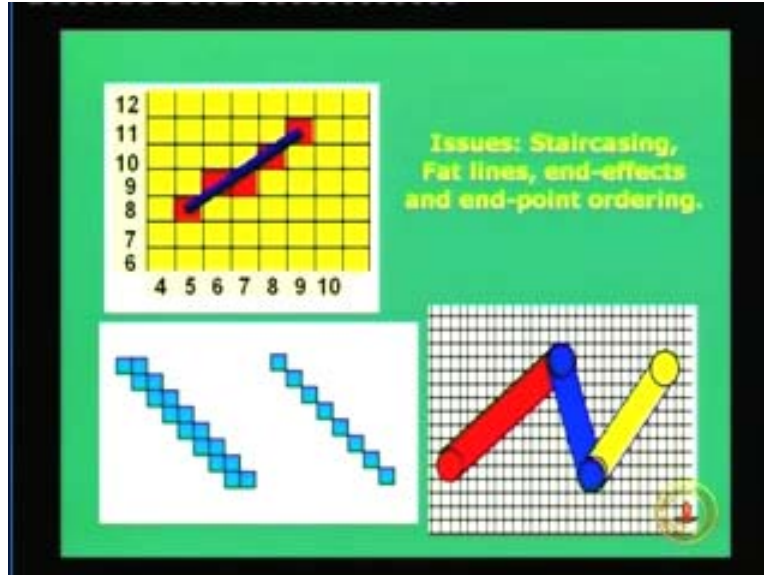
octant numbers four five to eight with you. But we have to see the section of the code which is involved in those two simple constructs or cases where you change the roles of x and y and draw from P₁ to P₀. Draw from P₁ to P₀ in terms of coding means swap P₀ to P₁ and decrement y or decrement delta dy as well.

(Refer Slide Time: 52:01)



So this is what is given here, draw from P₁ to P₀ the statement involves the following two codes swap P₀ P₁ that means make P₀ equal to P₁ and P₁ equal to P₀ and decrement y and make dy equal to minus dy. So that will automatically happen. So basically going down instead of going up and switch roles of x and y is swap x₁ y₁ swap x₀ y₀. It typically means that your x becomes y and y becomes x. You also swap the values of dx and dy which was used to initialize the variables and then you plot a point as y x instead of x, y. So once you do that automatically the octants are taking care of. These are the two cases for drawing a line in all the other quadrants. I will leave with the minimum amount of time left about the problem of aliasing. But before that I must tell you that you please discuss among yourselves and try to solve the problem of all the other octants. I discussed octant number one two and three only and I have given the statements for the octants four five six seven and eight in the slide just a few minutes back. And you should try to see with this role of reversing a P₀ P₁ reversing the role of x and y whether the statements are correct and you are able to draw the line correctly.

(Refer Slide Time: 54:08)



In addition to the problem of drawing a line in all the other octants the main issue was the effect of jaggies or stair casing. We have discussed this problem in the last class which is bound to happen in the digitized environment and this figure as you see in the slide here now I have shown pixels not as the intersection of grid points but within the squares. So, assuming you have a starting point at 5, 8 as like the previous problem and the finishing point is 9, 11 and we have calculated ourselves the following few points. And if these points in line ten you can see that there is a big effect of stair casing. So how you handle this effect of stair casing to meet the line appear smoother. That is the effect which we have to see, we will work on that the next class when we meet. So the issues are stair casing, you can have a problem of drawing flat lines which are either a line with one pixels wide or even two or three pixels or n pixels wide, you have problems of uniformity of drawing this. The other problem which I want to leave with you is the end point ordering which is also very very important.

End point ordering means if you want to draw one line one after the other and each of these lines have different colors, different widths and different shadings what about the end point? And in this case you say that you have drawn a red line then a blue line and then a yellow line what happens to the intersection points of these two lines? The starting and finishing points will have the red color and the yellow color but do you want the intersection point of the lines to have the color of the middle line or the red line or the yellow line. These are all important issues which one has to a thing about while design is algorithm in terms of implementation. Whichever gives the best view in terms of the entire figure we have to solve that. Of course we will discuss in the next class the effect of jaggies, stair casing or aliasing for drawing lines but mainly there are other issues involved in flat lines and also in end point ordering as to which one must take care when drawing lines. We will start with this slide when we meet in the next hour, till then good bye, thank you.