Image Representation

Course Code: CSC 3224 Course Title: Computer Graphics

Dept. of Computer Science
Faculty of Science and Technology

Lecturer No:	**3**	**Week No:**	**02**	**Semester:**	**Fall 2020-2021**
Lecturer:	**MAHFUJUR RAHMAN,** ***mahfuj@aiub.edu***				

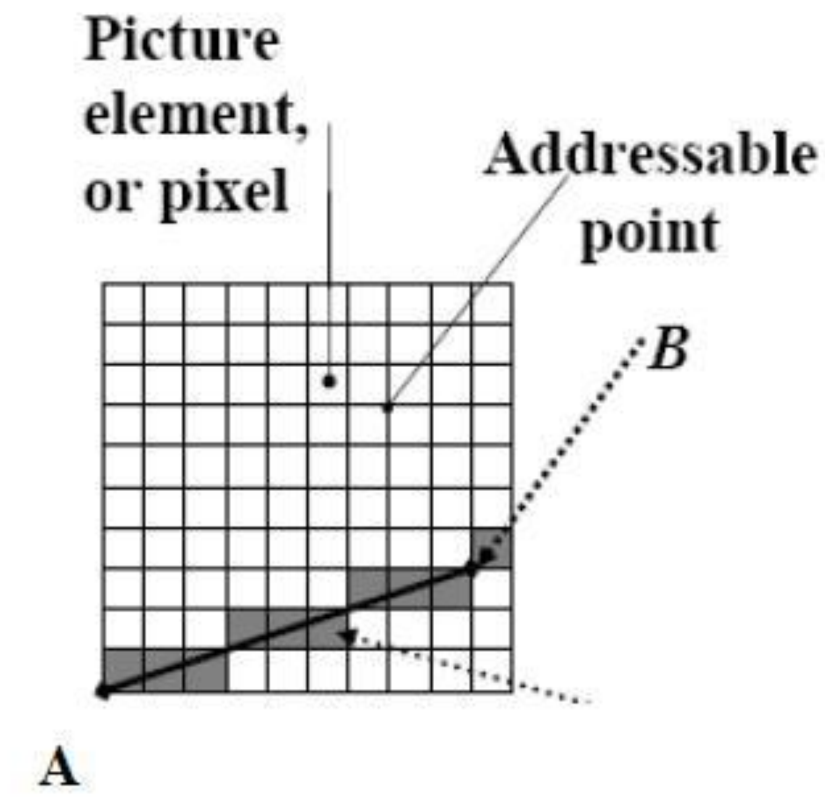
Outline

1. Pixel
2. Graphics Image
3. Color Model (RGB, CMY)
4. Direct Coding
5. Lookup Table
6. Display Monitor
7. Printing
8. Image Files
9. Books
10. References

Pixel

- A pixel is one of the many tiny dots that make up the representation of a picture in a computer's memory.
- Pixels in an image can be reproduced at any size without the appearance of visible dots or squares
- The intensity of each pixel is variable; in color systems, each pixel has typically three or four dimensions of variability such as red, green and blue, or cyan, magenta, yellow and black

Pixel



Computer Graphics Image

- Computer graphics can be created as either raster or vector images
  - **Raster Image**
  - **Vector Image**

Raster Image

- Raster graphics are **bitmaps**.

  - A bitmap is a grid of individual pixels that collectively compose an image.
  - Raster graphics render images as a collection of countless tiny squares.
  - Each square, or pixel, is coded in a specific shade. Individually, these pixels are worthless
  - Together, they’re worth a thousand words

Raster Image

❑ Usage of Raster Image:

- Raster graphics are best used for non-line art images; specifically digitized photographs, scanned artwork or detailed graphics

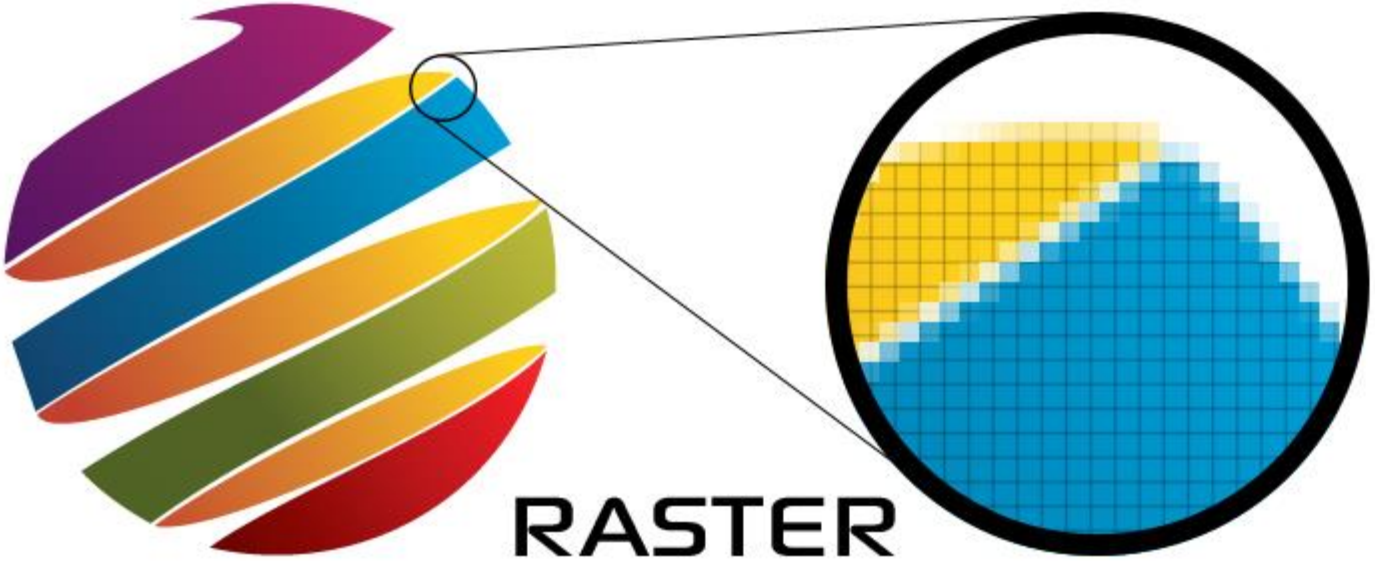
- Non-line art images are best represented in raster form because these typically include subtle chromatic gradations, undefined lines and shapes, and complex composition

Drawbacks of Raster Image

- Resolution in raster graphics is measured in **dpi,** or **dots per inch**. The higher the dpi, the better the resolution
- Raster files are significantly larger than comparable vector files, high resolution raster files are significantly larger than low resolution raster file
- Overall, as compared to vector graphics, raster graphics are less economical, slower to display and print, less versatile and more unwieldy to work with

Example of Raster Image

- ❑ Common raster formats include TIFF, JPEG, GIF, PCX and BMP files



Vector Image

- Unlike pixel-based raster images, vector graphics are based on **mathematical formulas** that define **geometric primitives** such as polygons, lines, curves, circles and rectangles

Use of Vector Image

- Because vector graphics are composed of true geometric primitives, they are best used to represent more structured images, like line art graphics with flat, uniform colors.
- Most created images meet these specifications, including logos, letterhead, and fonts.

Advantages of Vector Image

- vector-based graphics are more malleable than raster images
- They are much more versatile, flexible and easy to use
- The most obvious advantage of vector images over raster graphics is that vector images are quickly and perfectly scalable
- There is no upper or lower limit for sizing vector images

Advantages of Vector Image

- unlike raster graphics, vector images are not resolution-dependent
- Vector images have no fixed intrinsic resolution, rather they display at the resolution capability of whatever output device (monitor, printer) is rendering them
- because vector graphics need not memorize the contents of millions of tiny pixels, these files tend to be considerably smaller than their raster counterparts.

Vector Images

- Overall, vector graphics are more efficient and versatile.
- Common vector formats include AI, EPS, CGM, WMF and PICT (Mac).

Color Model

- A color model is a system for creating a full range of colors from a small set of primary colors.
- There are two types of color models: **additive** and **subtractive**.

Additive and Subtractive Model

- ❑ **Additive** color models **use light to display color**

✓while **Subtractive** color models **use printing inks**

- ❑ Colors perceived in additive models are the result of **transmitted light**

✓Colors perceived in subtractive models are the result of **reflected light**

RGB and CMYK

Color Model

- There are several established color models used in computer graphics, but the two most common are the **RGB model** (**Red**-**Green**-**Blue**) for computer display and the **CMYK model** (**Cyan**-**Magenta**-**Yellow**-**Black**) for printing.

Subtractive color (CMYK)

Additive Color (RGB)

RGB

- RGB uses additive color mixing, because it describes what kind of light needs to be emitted to produce a given color.

- Light is added together to create form from out of the darkness.

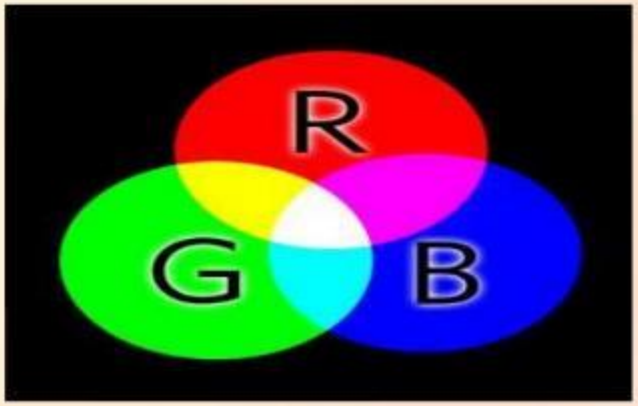
- RGB stores individual values for red, green and blue.

- (r,g,b)=>(0,0,0) black, (1,1,1) white [ranges 0 to 1]

RGB

RGB Color Model

- RGB is an additive color model For computer displays **uses light** to display color , Colors result from **transmitted light**
- Red + Green + Blue = White



RGB Value

- A color's RGB value indicates its **red, green,** and **blue** intensity.
- Each intensity value is on a scale of **0** to **255**

RGB Color Palette

3-bit RGB

Systems with a 3-bit RGB palette use 1 bit for each of the red, green and blue color components. That is, each component is either "on" or "off" with no intermediate states. This results in an 8-color $\mathbf{((2^1)^3 = 2^3 = 8)}$ palette

6-bit RGB

Systems with a 6-bit RGB palette use 2 bits for each of the red, green, and blue color components. This results in a 64-color $\mathbf{((2^2)^3 = 4^3 = 64)}$ palette

Color Palette

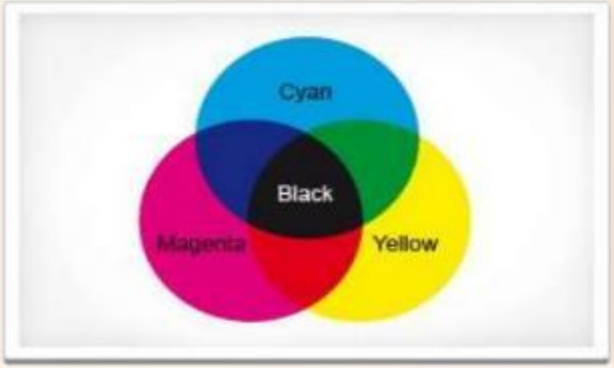
In computer graphics, a **color palette** is a finite set of colors. Palettes can be optimized to improve image accuracy in the presence of software or hardware constraints.

CMYK

CMYK Color Model

CMYK (subtractive color model) is the standard color model used in offset printing for full-color documents. Because such printing uses inks of these four basic colors, it is often called **four-color printing**.

- Where two colors of RGB overlaps, we see a new color formed by mixing of the two additive primaries. These new colors are:
- A greenish blue called **cyan**.
- A blushed red called **magenta**.
- A bright **yellow**.
- The key color **, Black**.

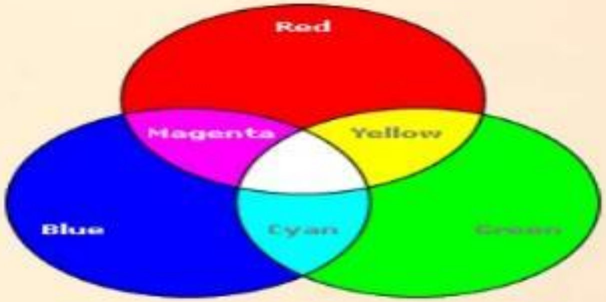


CMYK

CMYK Color Model

We can express this effect pseudo-algebraically. Writing **R**, **G** and **B** for **red, green** and **blue**, **C**, M and **Y** for **cyan**, magenta and **yellow**, and **W** for **white**, and using (+) to mean additive mixing of light, and (–) to mean subtraction of light, we have:

- C (cyan) = **G** + **B** = **W** - **R**
- M (magenta) = **R** + **B** = **W** - **G**
- Y (yellow) = **R** + **G** = **W** – **B**



In each equation, the colour on the left is called the **complementary** colour of the one at the extreme right; for example, magenta is the complementary colour of **green**.

CMY

- CMY uses subtractive color mixing used in the **printing process**, because it describes what kind of inks need to be applied so the light reflected from the substrate and through the inks produces a given color.

- One starts with a white substrate (canvas, page, etc.), and uses ink to subtract color from white to create an image.

- CMYK stores ink values for cyan, magenta, yellow, key(Black).

- CMYK(c%, m%, y%)=>(0%,0%,0%) white. [ranges from 0 to 100%]

RGB to CMY

- C = 1 – (color.R / 255.0);
- M = 1 – (color.G / 255.0);
- Y = 1 – (color.B / 255.0);

CMY to RGB

- R = (1 - C) * 255.0,
- G = (1 - M) * 255.0,
- B = (1 - Y) * 255.0

RGB -> CMY -> RGB

More info

http://colormine.org/convert/rgb-to-cmy

Sample Code:

https://github.com/THEjoezack/ColorMine/blob/master/ColorMine/ColorSpaces/Conversions/CmyConverter.cs

Direct Coding

- Basically, images are the collections of several pixels with colors. In computer graphics, direct coding is an algorithm that provides some amount of storage space for each pixel so that the pixel is coded with a color.

Direct Coding

- ❑ Storage space for each pixel to code the color
- ❑ Use 3 bits per pixel (1 for R, 1 for G and 1 for B) [Industry standard]
- ❑ 256 different intensity level for each color

Bit 1 R	Bit 2 G	Bit 3 B	Color Name
0	0	0	Black
0	0	1	Blue
0	1	0	Green
1	0	0	Red

Direct Coding

Basically images are the collections of several pixels with colors. In computer graphics, direct coding is an algorithm that provides some amount of storage space for each pixel so that the pixel is coded with a color.

Direct Coding of colors with 3 bits:

Red Bit 1	Green Bit 2	Blue Bit 3	Color Name
0	0	0	Black
0	0	1	Blue
0	1	0	Green
0	1	1	Cyan
1	0	0	Red
1	0	1	Magenta
1	1	0	Yellow
1	1	1	White

For color images, the common standard used for filling the pixel uses 3-bit colors or 24-bit per pixel. A primary color has 256 intensity levels, which is equal to the binary values starting from 00000000 to 1111111. Hence, a pixel can take a color from 256 × 256 × 256 possibilities (16.7 million) that is 256 for each of red, green, and blue colors. The 24-bit format is commonly referred to as true color representation.

The Black, white, gray scale image representations is the most common aspect of direct coding. Black and white images require one bit per pixel, here the bit value 0 represents black and bit value 1 represents white. The gray scaled images are coded with the help of 8 bits per pixel, to allow a total of 256 intensity levels. Even though direct coding method supports a lot of applications, there is shortage of storage space with the 24-bit standard. Usually in an application, the total number of colors in any image is very less. Hence, for a 24-bit representation to have 16.7 million colors in an image will be overloaded.

Lookup Table

In computer graphics, lookup tables are used to store the starting addresses of each line and the values corresponding to the placement of pixels within a byte.

Steps to plot a point using lookup table

1. Locate the starting address corresponding to the line on which the point is to appear.

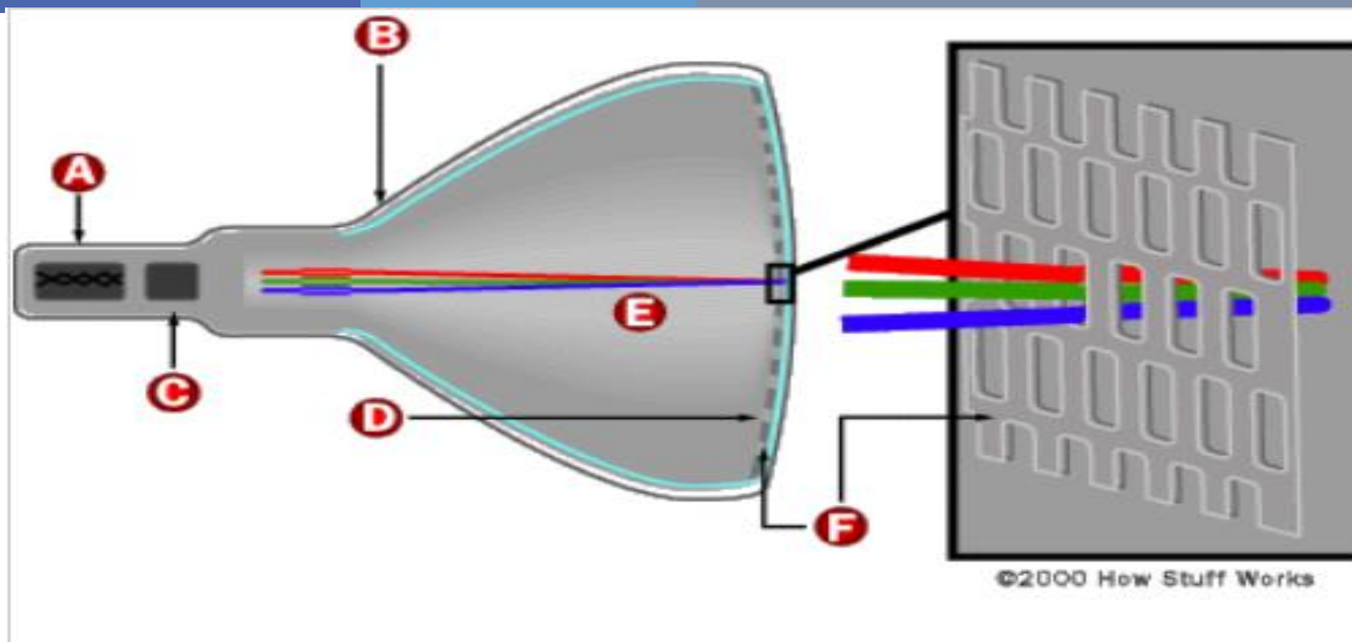
2. Locate the address of the byte in which the point will be represented.

Lookup Table

- Pixel values do not code colors directly
- Refer to a table of color values
- A table with 256 colors with RGB values

o	r	g	b
1	11111111	11111111	11111111
2			
255			

Display Monitor (CRT)



- **A** Cathode
- **B** Conductive coating
- **C** Anode
- **D** Phosphor-coated screen
- **E** Electron beams
- **F** Shadow mask

Printing

- ❑ Halftone
- ❑ Go through chapter 2 (schaum’s outline) for details.

Halftone

- **Halftone** is the technique that simulates continuous tone imagery through the use of dots.
- Dots can be varied either
  - in size
  - in shape or
  - in spacing
- Halftone generates a gradient like effect.

Halftone Image

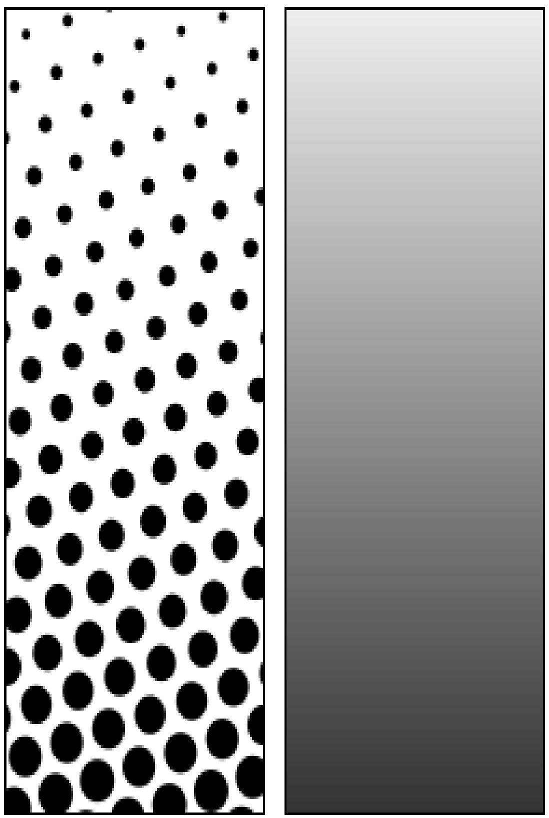
- A halftone, or halftone image, is an image comprised of discrete dots rather than continuous tones. When viewed from a distance, the dots blur together, creating the illusion of continuous lines and shapes.

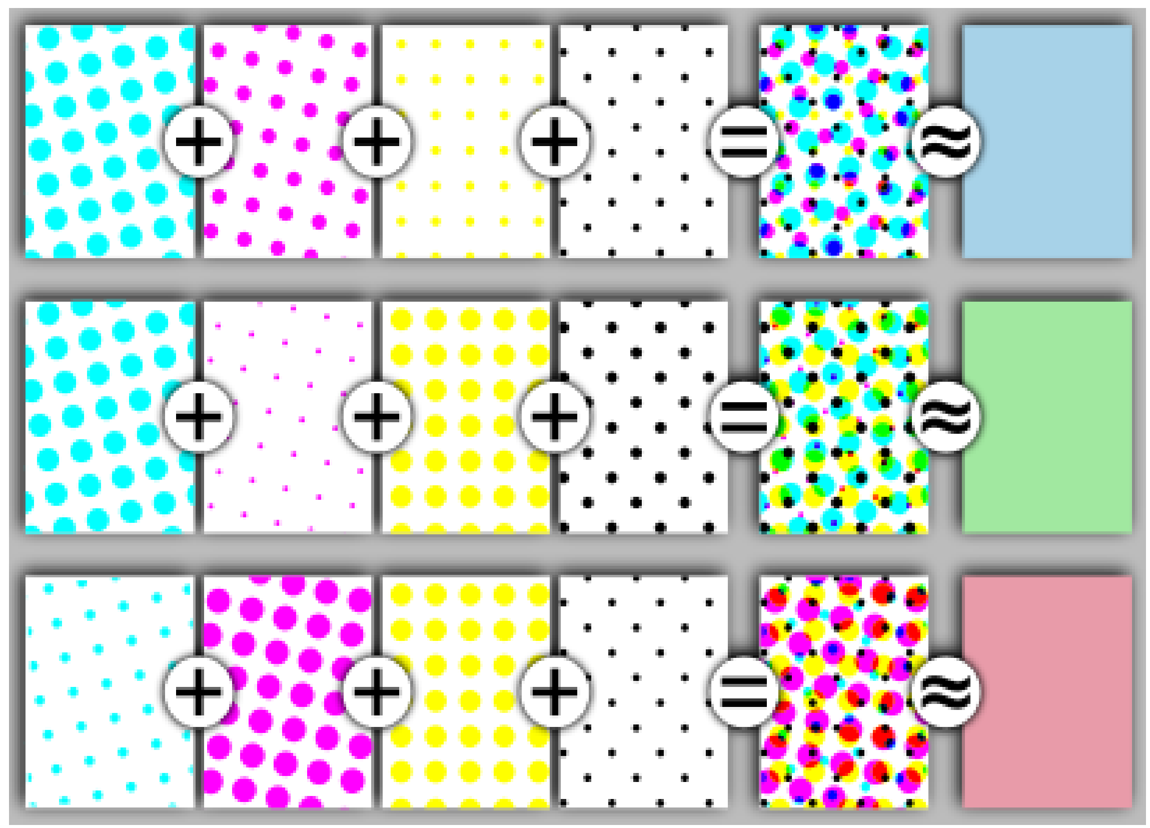
- ✓ By halftoning an image (converting it from a bitmap to a halftone), it can be printed using less resources

How Halftone work

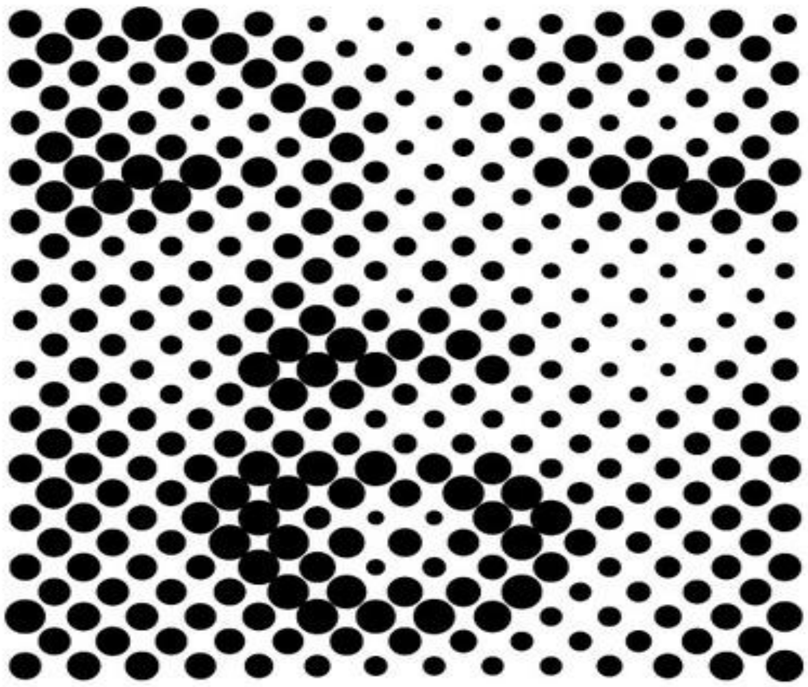
- Halftone process, in printing, a technique of breaking up an image into a series of dots so as to reproduce the full tone range of a photograph or tone art work.

Example

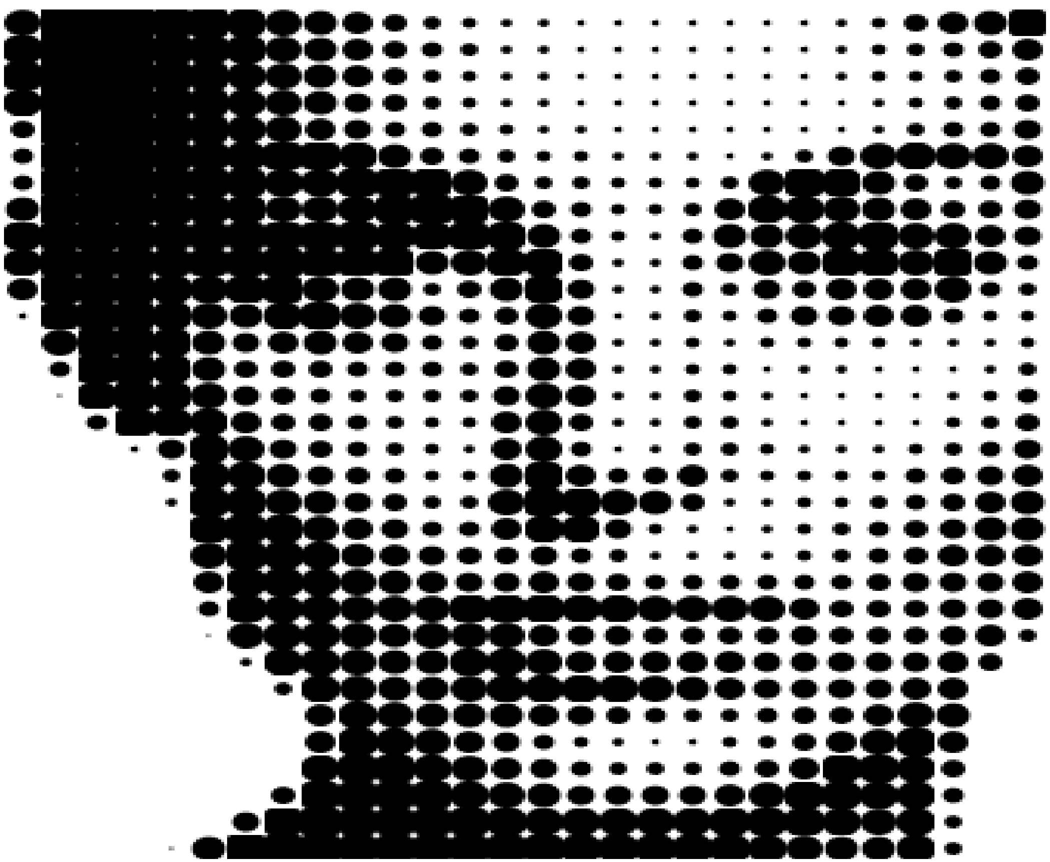




Contd.



Contd.



Books

- Foley, van Dam, Feiner, Hughes, Computer Graphics: principles and practice, Addison Wesley, Second Edition.
- Schaum's Outline of Theory & Problems of Computer Graphics.
- Peter Shirley Steve Marschner , “Fundamental of computer graphics”, Third Edition.

References

- http://colormine.org/convert/rgb-to-cmy
- www.howstuffworks.com
- www.wikipedia.com
- http://www.picturetopeople.org/image_effects/photo-halftone/examples/photo-to-halftone-convertion-2.gif
- http://mocoloco.com/fresh2/upload/2011/12/halftone_calendar_by_casey_klebba/halftone_calendar_casey_klebba_3b-thumb-468x468-35319.jpg
- https://www.chegg.com
- https://www.slideshare.net/mustafasalam167/color-model-29181025
- https://www.printcnx.com/resources-and-support/addiational-resources/raster-images-vs-vector-graphics/
- https://slideplayer.com/slide/5143930/

Scan Conversation (Part 1)

Course Code: CSC 3224 Course Title: Computer Graphics

Dept. of Computer Science
Faculty of Science and Technology

Lecturer No:	**4**	**Week No:**	**03**	**Semester:**	**Fall 2020-2021**
Lecturer:	**MAHFUJUR RAHMAN,** ***mahfuj@aiub.edu***				

Lecture Outline

1. What is Scan Conversation?
2. What is Rasterisation?
3. What is Incremental Algorithm?
4. Characteristics of Incremental Algorithm.
5. DDA Line Incremental Algorithm (Derivation)
6. DDA Line Incremental Algorithm (Mathematics)
7. Disadvantages of DDA Line Incremental Algorithm
8. Midpoint Line Algorithm (Derivation)
9. Midpoint Line Algorithm (Mathematics)
10. Advantages of Midpoint Line Algorithm

Scan Conversation

- Final step of rasterisation (process of taking geometric shapes and converting them into an array of pixels stored in the framebuffer to be displayed)
- Takes place after clipping occurs
- All graphics packages do this at the end of the rendering pipeline
- Takes triangles and maps them to pixels on the screen
- Also takes into account other properties like lighting and shading, but we'll focus first on algorithms for line scan conversion

Rasterisation

Rasterisation (or rasterization) is the task of taking an image described in a vector graphics format (shapes) and converting it into a raster image (pixels or dots) for output on a video display or printer, or for storage in a bitmap file format.

Incremental Algorithm

- Incremental Algorithm is a line drawing algorithm

✓ Start from initial pixel to reach final pixel to draw a line

Incremental Algorithm

Finding the next pixel

Special cases:

- **Horizontal Line:**
  - Draw pixel P and increment x coordinate value by 1 to get next pixel.
- **Vertical Line:**
  - Draw pixel P and increment y coordinate value by 1 to get next pixel.
- **Diagonal Line:**
  - Draw pixel P and increment both x and y coordinate by 1 to get next pixel.
- What should we do in general case?
  - Increment x coordinate by 1 and choose point closest to line.
  - But how do we measure "closest"?

Incremental Algorithm

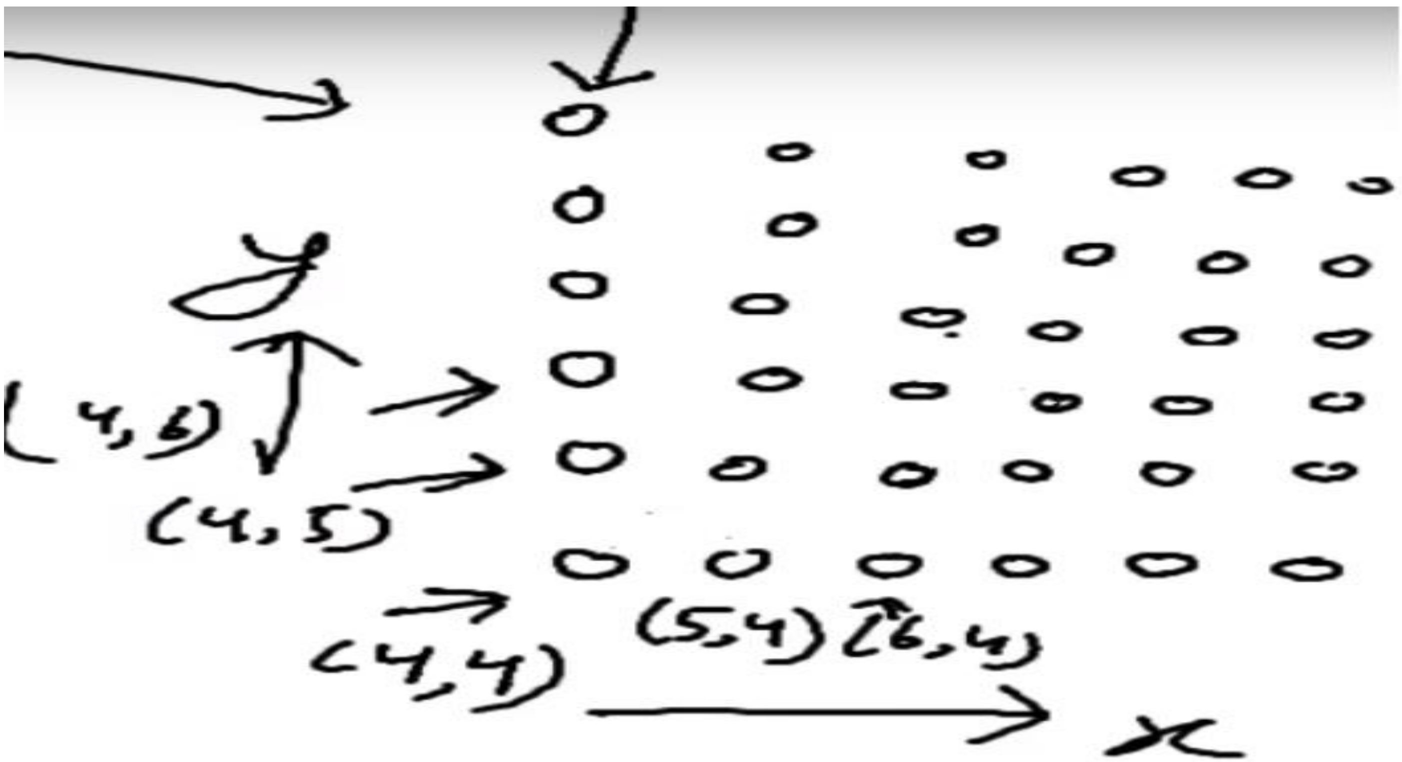
Requirements

- Three Requirements:

- Integer Pixel Grid
- Slope
- Disjoint Pixel

Incremental Algorithm

Integer Pixel Grid



Incremental Algorithm

Slope

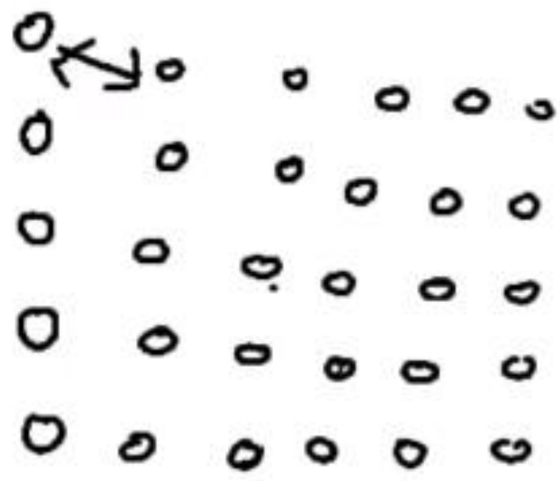
(x, y) (x_1, y_1)

$$m = \frac{y_1 - y}{x_1 - x}$$

$$m \leq 1$$

Incremental Algorithm

Disjoint Pixel



Digital Differential Analyzer (DDA)

Bresenham's Incremental Algorithm

In computer graphics, a digital differential analyzer (DDA) is hardware or software used for interpolation of variables over an interval between start and end point. DDAs are used for rasterization of lines, triangles and polygons.(graphics algorithm)

Digital Differential Analyzer (DDA)

Derivation

BIA

(x_n, y_n)

(x_1, y_1)

$$m = \frac{y_n - y_1}{x_n - x_1}$$

$$y = mx + B$$

$$(x_i, y_i)$$

$$x_i = x_1$$
$$y_i = y_1$$

*** M =Slope**

***B=How far line from center**

Digital Differential Analyzer (DDA)

Derivation (if m<1)

1st Pixel:

$$x_i = x_1$$
$$y_i = y_1$$

Next Pixel:

$$x_{i+1} = x_i + \Delta x$$
$$y_{i+1} = m x_{i+1} + B$$

Digital Differential Analyzer (DDA)

Derivation

$$x_{i+1} = x_i + \Delta x$$

$$y_{i+1} = m x_{i+1} + B$$

$$= m(x_i + \Delta x) + B$$

$$= \underbrace{m x_i + B} + m\Delta x$$

$$= y_i + m \underline{\underline{\Delta x}} \qquad \Delta x = 1$$

$$y_{i+1} = y_i + m$$

$$\{x_{i+1}, \ \text{Round}(y_{i+1})\}$$

Digital Differential Analyzer (DDA)

Derivation (if m<1)

Continue the Process:

$$\{x_{i+1},\ \text{Round}(y_{i+1})\}$$

Until Acheived:

$$x_{i+1} = x_n$$

Digital Differential Analyzer (DDA)

Derivation (if m<1)

$$\{x_{i+1},\ \text{Round}(y_{i+1})\}$$

$$\text{Round}(y_{i+1}) = \lfloor y_{i+1} + 0.5 \rfloor$$

Digital Differential Analyzer (DDA)

Derivation (if m>1 and m=1)

if m>1

x1 = round(x + 1 /m)

y1 = y + 1

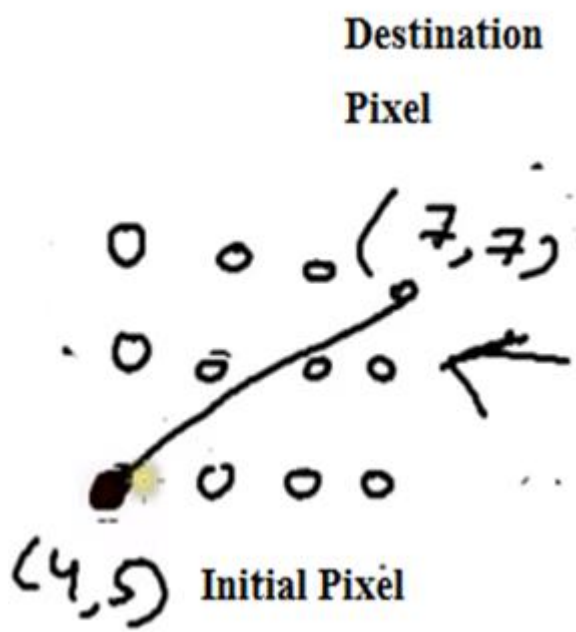
if m=1

x1=x+1

y1=y+1

Digital Differential Analyzer (DDA)

Mathematics (Question)



Digital Differential Analyzer (DDA)

Mathematics (Solve): Calculate Slope m and 1st Pixel

Slope:

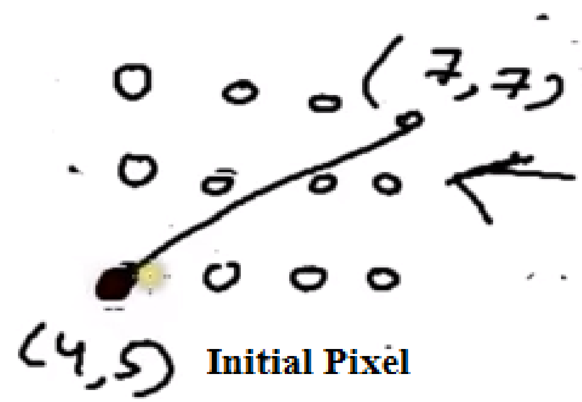
$$m = \frac{7-5}{7-4} = 0.67$$

Initial Pixel

$$x_i = 4$$

$$y_i = 5$$

Destination Pixel



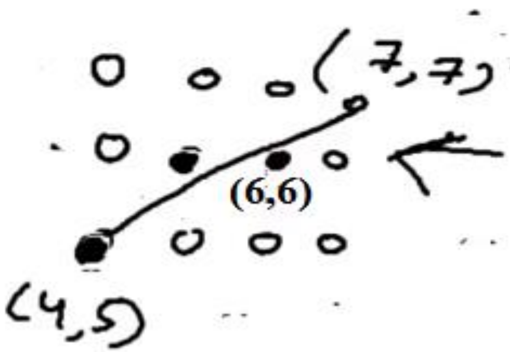
Initial Pixel

Digital Differential Analyzer (DDA)

Mathematics (Solve): Calculate 3rd pixel

$x_{i+2} = 5+1 = 6$ (6,6)

$y_{i+2} = 5.67+0.67 = 6.34$



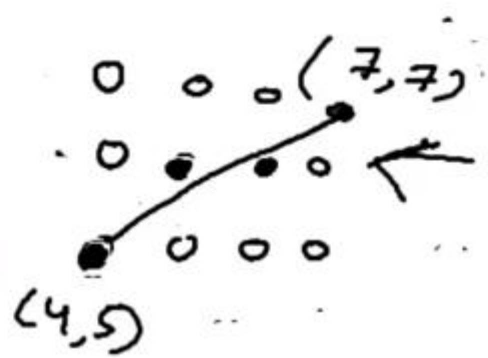
Digital Differential Analyzer (DDA)

Mathematics (Solve): Calculate Final pixel

$$x_{i+3} = 6+1 = 7$$

$$y_{i+3} = 6.34 + 0.67 = 7.01$$

(7,7)



Digital Differential Analyzer (DDA)

Disadvantages

- Floating point arithmetic in DDA algorithm is still time consuming. The algorithm is orientation dependent. Hence end point accuracy is poor.

- Although DDA is fast, the accumulation of round-off error in successive additions of floating point increment, however can cause the calculation pixel position to drift away from the true line path for long line segment

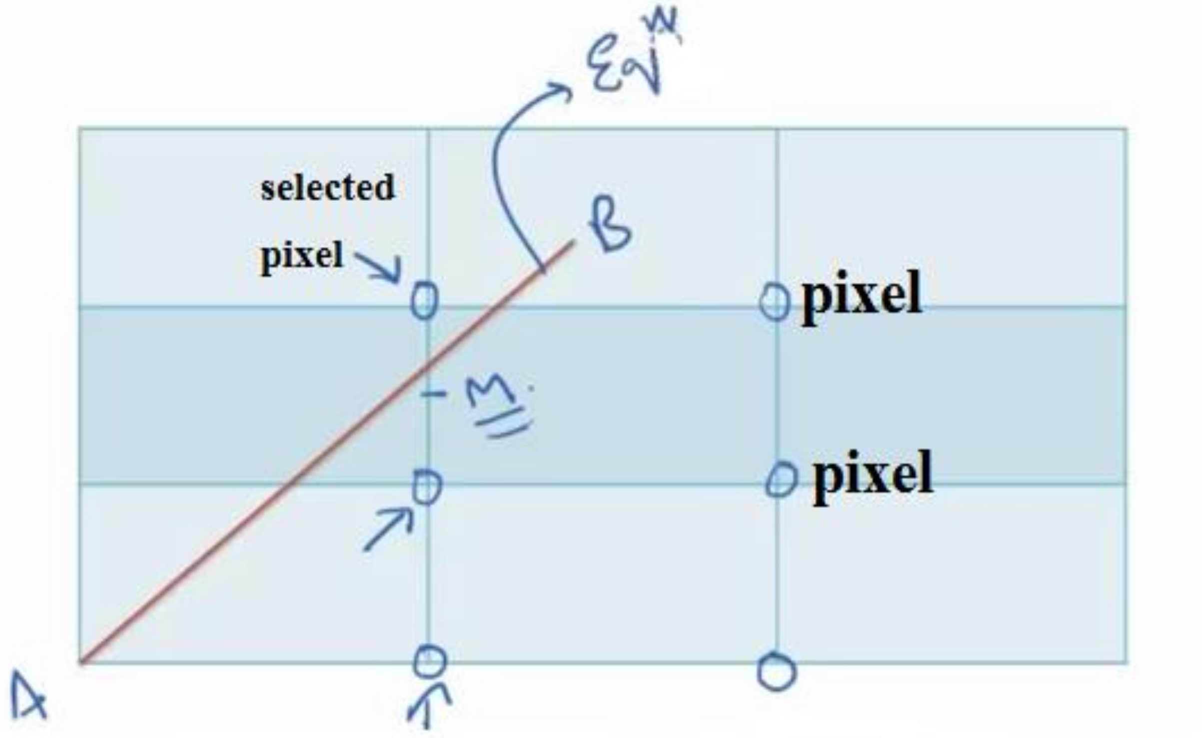
Bresenham's Mid Point Line Algorithm

Introduction

The Midpoint line algorithm is an incremental line plotting algorithm i.e. at each step we make incremental calculations based on preceding step to find next y value, in order to form a close approximation to a straight line between two points.

Bresenham's Mid Point Line Algorithm

Concept



Bresenham's Mid Point Line Algorithm

Concept

Equation of Line:

$y=mx+B$

Function of Line:

$F(x,y)= ax+by+c$

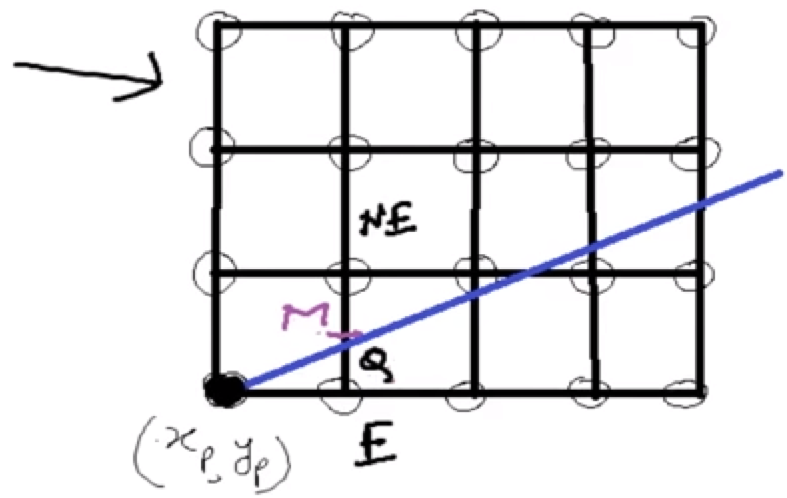
Bresenham's Mid Point Line Algorithm

Derivation

Mid Point line algorithm

$dy = y_n - y_0$

$dx = x_n - x_0$



$f(x,y) = ax + by + c = 0$

$y = mx + B$

$= \frac{dy}{dx} x + B$

$\Rightarrow \quad dy \cdot x - dx \cdot y + B\,dx = 0$

Bresenham's Mid Point Line Algorithm

Derivation

$$y = \underbrace{m}_{\text{slope}}x + \underbrace{B}_{\text{y-intercept}}.$$

$$m = \frac{dy}{dx} \qquad y = \left(\frac{dy}{dx}\right)x + B.$$

$$F(x,y) = ax + by + c = 0.$$

$$F(x,y) = dy(x) + (-dx)y + B.dx = 0$$

$$\left\{\begin{array}{l} a = dy \\ b = -dx \\ c = B.dx \end{array}\right\}$$

Bresenham's Mid Point Line Algorithm

Derivation

Decision Parameter

$d = F(x, y)$

at M, $d = (x_p + 1,\ y_p + \frac{1}{2})$

$\Rightarrow a.(x_p + 1) + b(y_p + \frac{1}{2}) + c$

$d_{old} = \underline{\underline{d}};$

Based on the sign of $d \rightarrow$ select E OR NE

Bresenham's Mid Point Line Algorithm

Derivation (Find Dstart)

$$d_{start} = f\left(x_0+1,\ y_0+\tfrac{1}{2}\right)$$

xo,y0 first pixel

$$= a(x_0+1) + b\left(y_0+\tfrac{1}{2}\right) + c$$

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

$$= \underline{f(x_0, y_0)} + a + b/2$$

$$= a + b/2$$

$\underline{f(x_0, y_0)}$ **=o as the pixels on the line**

Bresenham's Mid Point Line Algorithm

Derivation (Find Dstart)

$= a + b/2$

$= dy + (-dx)/2$

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

$\cancel{2}$ $d_{start} = 2dy - dx$

$d > 0$

$\frac{d}{2} > 0$

$d * 2 > 0$

Bresenham's Mid Point Line Algorithm

Derivation (How to Choose E / NE)

$d > 0$ NE is chosen

$d < 0$ E " "

Bresenham's Mid Point Line Algorithm

Derivation (If E is chosen)

$$d_{new} = F\left(x_p+2,\ y_p+\frac{1}{2}\right)$$

$$d_{new} - d_{old} = f\left(x_p+2,\ y_p+\frac{1}{2}\right) - f\left(x_p+1,\ y_p+\frac{1}{2}\right)$$

$$= a(x_p+2) + b\left(y_p+\frac{1}{2}\right) + c - a(x_p+1) - b\left(y_p+\frac{1}{2}\right) - c$$

Bresenham's Mid Point Line Algorithm

Derivation (If E is chosen)

$$d_{new} - d_{old} = f\left(x_p+2, y_p+\frac{1}{2}\right) - f\left(x_p+1, y_p+\frac{1}{2}\right)$$

$$= a(x_p+2) + b\left(y_p+\frac{1}{2}\right) + c - a(x_p+1) - b\left(y_p+\frac{1}{2}\right) - c$$

$$= a$$

so $d_{new} = d_{old} + a$

Bresenham's Mid Point Line Algorithm

Derivation (If NE is chosen)

if NE is chosen

$$d_{new} = f(x_p + 2, y_p + 3/2)$$

$$d_{new} - d_{old} = a + b$$

$$d_{new} = d_{old} + a + b$$

Bresenham's Mid Point Line Algorithm

Derivation (Adjustment with value 2)

For E : $d_{new} - d_{old} = a \cong 2a = 2dy$

For NE : $d_{new} - d_{old} = a + b = 2(a+b)$

$= 2(dy - dx)$

Bresenham's Mid Point Line Algorithm

Derivation (Summary)

$$\left\{\begin{array}{l} d_{start} = 2dy - dx \\ (\Delta d)_E = 2dy \\ (\Delta d)_{NE} = 2(dy - dx) \end{array}\right\}$$

Δd : New - Old

$d > 0$ NE is chosen

$d <= 0$ E " "

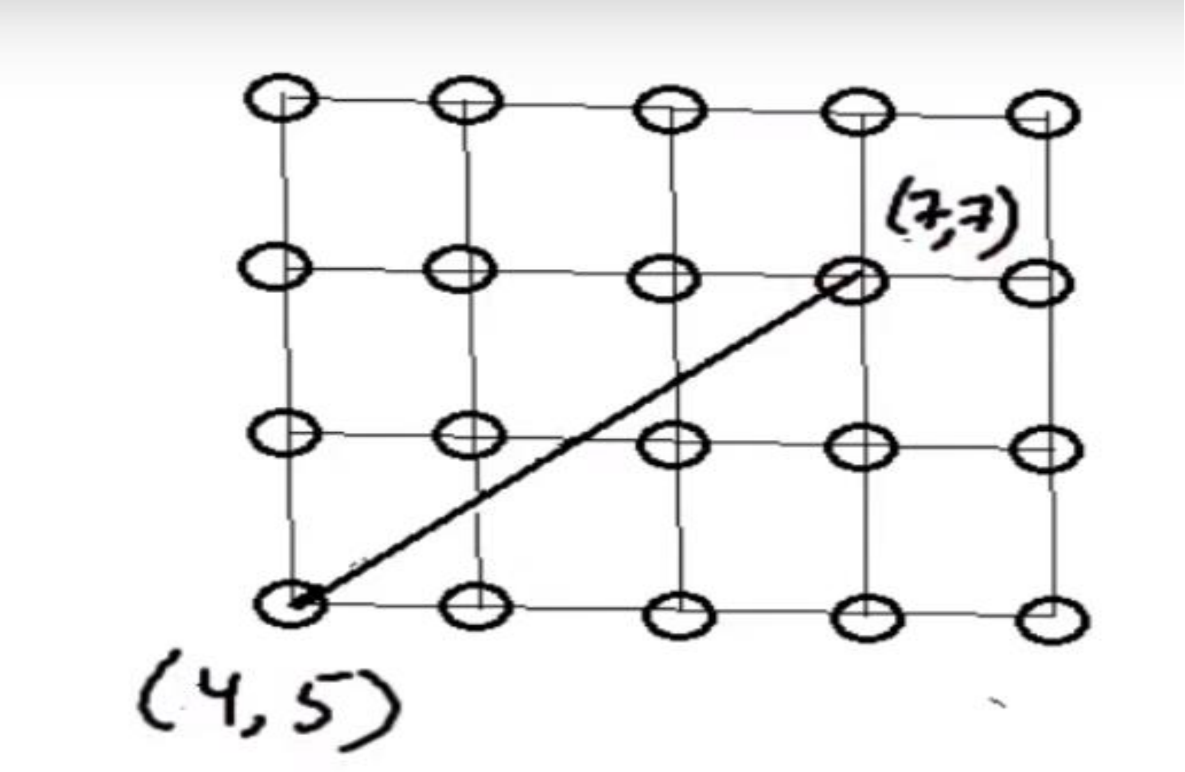
Bresenham's Mid Point Line Algorithm

Advantages

- Incremental Method
- No Round Function
- More accurate position
- Only Arithmetic Function

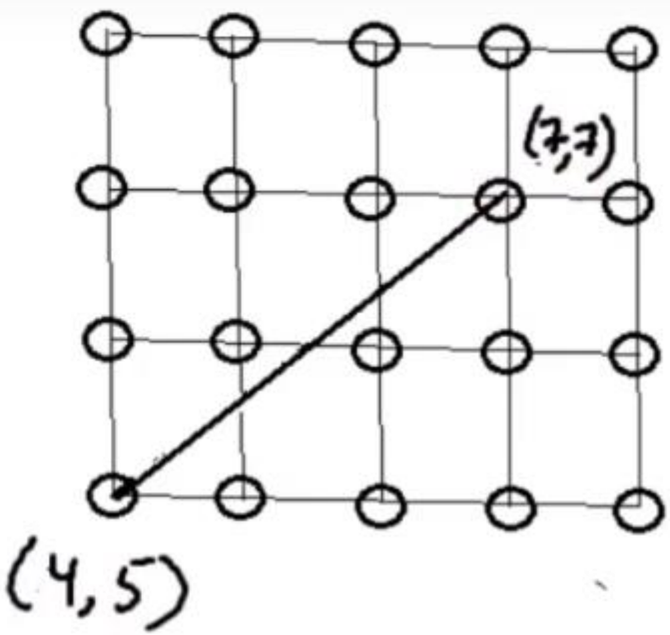
Bresenham's Mid Point Line Algorithm

Mathematics : Question



Bresenham's Mid Point Line Algorithm

Mathematics: Solve



$(x_0, y_0) = (4, 5)$

$(x_n, y_n) = (7, 7)$

Bresenham's Mid Point Line Algorithm

Mathematics

$$(x_0, y_0) = (4, 5)$$

$$(x_n, y_n) = (7, 7)$$

$$dy = y_n - y_0 = 7 - 5 = 2$$

$$dx = x_n - x_0 = 7 - 4 = 3$$

$$d_{start} = 2dy - dx = 2 \times 2 - 3 = 1$$

$$2dy = 2 \times 2 = 4$$

$$2(dy - dx) = 2(2 - 3) = -2$$

Bresenham's Mid Point Line Algorithm

Mathematics

$(x_n, y_n) = (7, 7)$

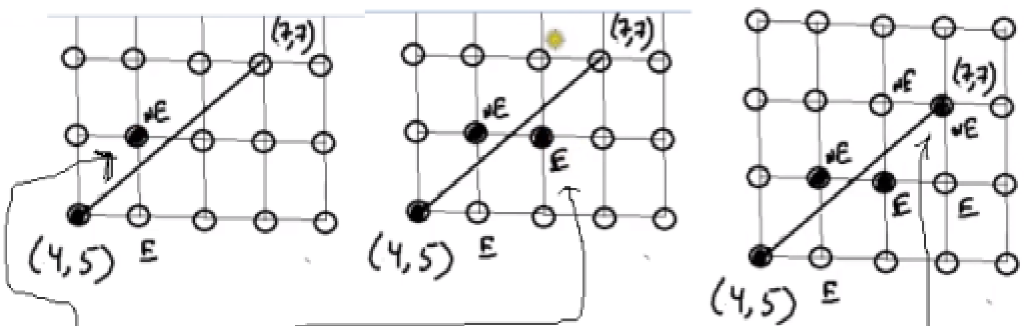
$dy = y_n - y_0 = 7 - 5 = 2$

$dx = x_n - x_0 = 7 - 4 = 3$

$d_{start} = 2dy - dx = 2 \times 2 - 3 = 1$

$2dy = 2 \times 2 = 4$

$2(dy - dx) = 2(2-3) = -2$

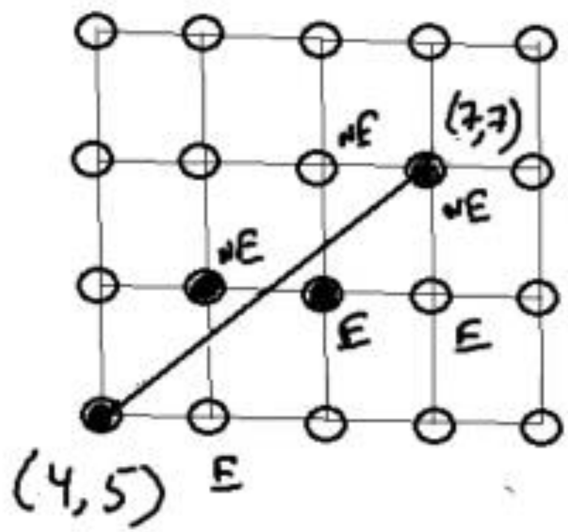


d	Compare With 0	Decision	x	y
1	$d > 0$	NE	5	6
$1-2=-1$	$-1 < 0$	E	6	6
$-1+4=3$	$3 > 0$	NE	7	7

Bresenham's Mid Point Line Algorithm

Mathematics

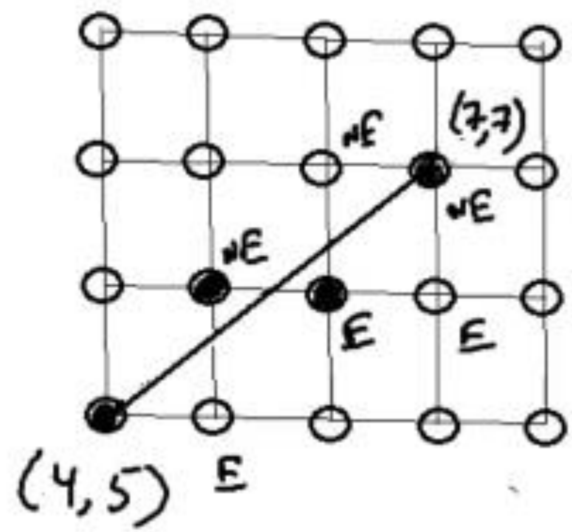
x	y
5	6
6	6
7	7



Bresenham's Mid Point Line Algorithm

Mathematics

x	y
5	6
6	6
7	7



Scan Conversation (Part 1)

Course Code: CSC 3224 Course Title: Computer Graphics

Dept. of Computer Science
Faculty of Science and Technology

Lecturer No:	**5**	**Week No:**	**04**	**Semester:**	**Fall 2020-2021**
Lecturer:	**MAHFUJUR RAHMAN,** ***mahfuj@aiub.edu***				

Lecture Outline

1. Midpoint Circle Algorithm (Derivation)
2. Midpoint Circle Algorithm (Mathematics)

Bresenham's Mid Point Circle Algorithm

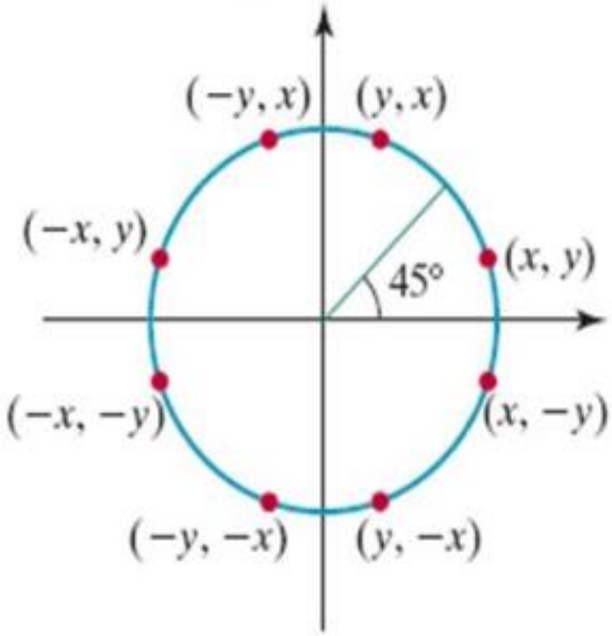
Introduction

In computer graphics, the midpoint circle algorithm is an algorithm used to determine the points needed for rasterizing a circle. Bresenham's circle algorithmis derived from the midpoint circle algorithm

Bresenham's Mid Point Circle Algorithm

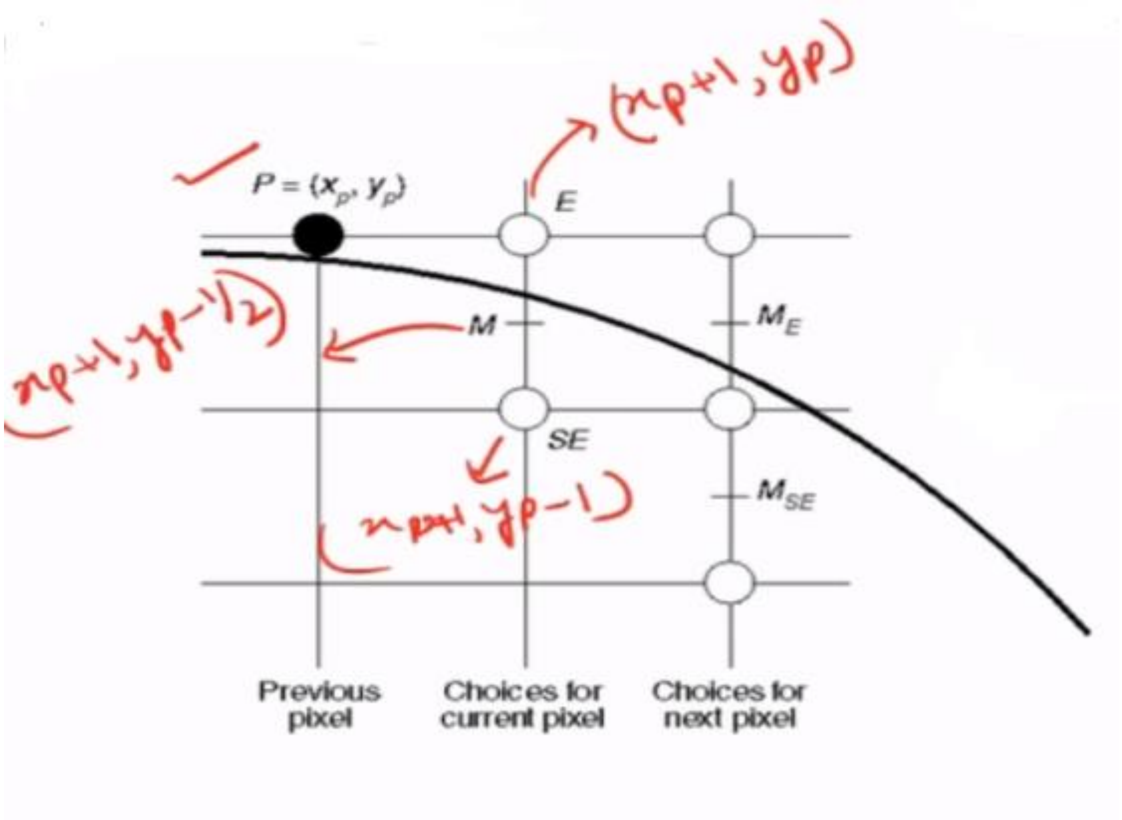
Concept

- **Using 8-way symmetry**



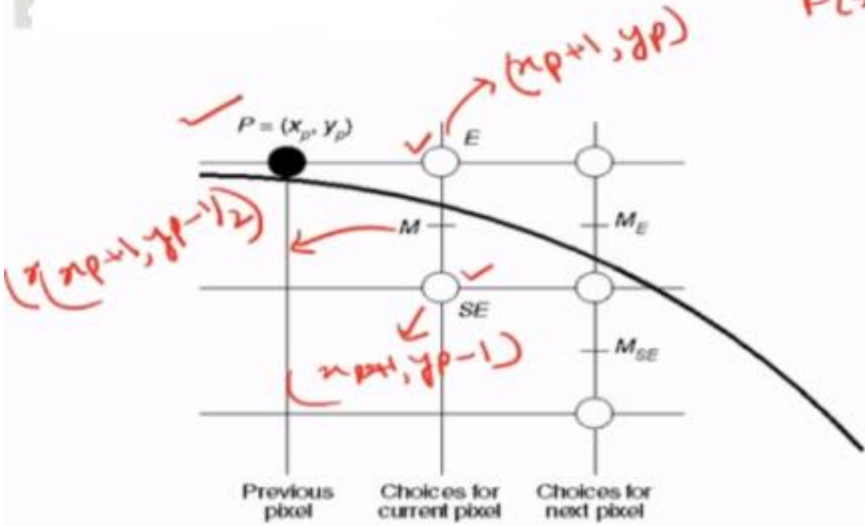
Bresenham's Mid Point Circle Algorithm

Concept



Bresenham's Mid Point Circle Algorithm

Concept



$F(x,y) = x^2 + y^2 - R^2 = 0$

$\boxed{E} / \boxed{SE} \longrightarrow M$ (Mid Point)

$F(x,y) > 0 \rightarrow$ point outside the circle

$F(x,y) < 0 \rightarrow$ point inside the circle.

Bresenham's Mid Point Circle Algorithm

Derivation

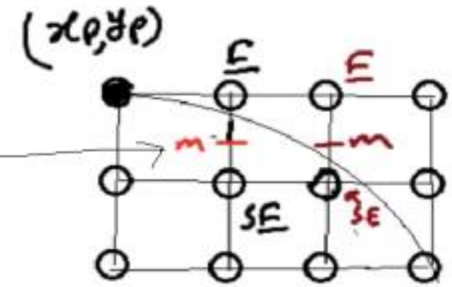
Mid Point circle

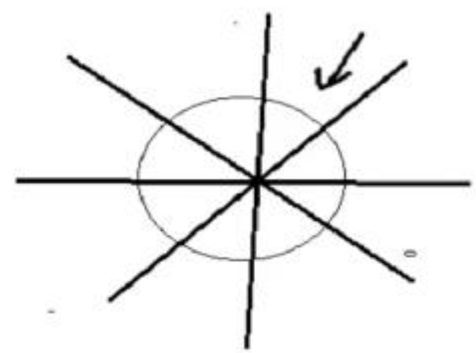
$$x^2 + y^2 - R^2 = 0$$

$$f(x,y) = x^2 + y^2 - R^2$$

$$(x,y)$$

$$d = f\left(x_p + 1, y_p - \frac{1}{2}\right)$$





Bresenham's Mid Point Circle Algorithm

How to chose E/SE

$d = F(M).$

$d >= 0 \Rightarrow \boxed{SE}$

$d < 0 \Rightarrow \boxed{E}$

Bresenham's Mid Point Circle Algorithm

Find dstart

$$d_{start} = F(M)$$
$$= F(x_0+1,\ y_0-\tfrac{1}{2}) = F(1,\ R-\tfrac{1}{2}).$$

$$d_{start} = f(0+1,\ R-\tfrac{1}{2})$$
$$= 1^2 + (R-\tfrac{1}{2})^2 - R^2$$
$$= \frac{5}{4} - R$$
$$= 1 - R \quad [\,5/4 \cong \underline{1}\,]$$

Bresenham's Mid Point Circle Algorithm

if E Selected

Pixel E

$$d_{new} = f\left(x_p + 2,\ y_p - \tfrac{1}{2}\right)$$

$$\Delta_E = d_{new} - d_{old}$$

$$= f\left(x_p + 2,\ y_p - \tfrac{1}{2}\right) - f\left(x_p + 1,\ y_p - \tfrac{1}{2}\right)$$

$$= (x_p + 2)^2 + \underline{\left(y_p - \tfrac{1}{2}\right)^2} - \underline{R}^2 - (x_p + 1)^2 - \underline{\left(y_p - \tfrac{1}{2}\right)^2} + \underline{R}^2$$

$$= x_p^2 + 4x_p + 4 - x_p^2 - 2x_p - 2$$

$$= 2x_p + 3$$

Bresenham's Mid Point Circle Algorithm

if SE Selected

selected SE

$$d_{new} = f\left(x_p+2,\ y_p-\frac{3}{2}\right)$$

$$\Delta_{SE} = f\left(x_p+2,\ y_p-3/2\right) - f\left(x_p+1,\ y_p-\frac{1}{2}\right)$$

$$= 2x_p - 2y_p + 5$$

Bresenham's Mid Point Circle Algorithm

Finally

$d >= 0$ SE is selected

$d < 0$ E is "

$d_{start} = 1 - R$

$\Delta_E = 2x_p + 3$

$\Delta_{SE} = 2x_p - 2y_p + 5$

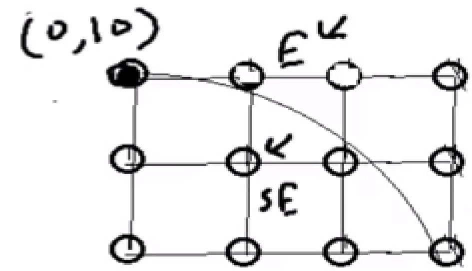
Bresenham's Mid Point Circle Algorithm

Mathematics (Question) : Center 0,0

Midpoint circle Example

10 — Radius

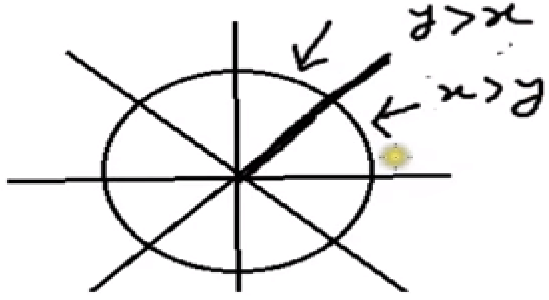
Given



$d_{start} = 1 - R = 1 - 10 = -9$

$\Delta_E = 2x_p + 3$

$\Delta_{SE} = 2(x_p - y_p) + 5$



Bresenham's Mid Point Circle Algorithm

Mathematics (solution)

d	comparison	Decision	x	y
			0	10
-9	$d<0$	E	1	10
$-9+2*0+3=-6$	$d<0$	E	2	10
$-6+2*1+3=-1$	$d<0$	E	3	10
$-1+2*2+3=6$	$d>0$	SE	4	9

Bresenham's Mid Point Circle Algorithm

Mathematics (solution)

$-9+2*0+3 = -6$	$d<0$	E	2	10
$-6+2*1+3=-1$	$d<0$	E	3	10
$-1+2*2+3=6$	$d>0$	SE	4	9
$6+2(3-10)+5 = -3$	$d<0$	E	5	9
$-3+2*4+3= 8$	$d>0$	SE	6	8

Bresenham's Mid Point Circle Algorithm

Mathematics (solution)

$-1+2*2+3=6$	$d>0$	SE	4	9
$6+2(3-10)+5 = -3$	$d<0$	E	5	9
$-3+2*4+3= 8$	$d>0$	SE	6	8
$8+2(5-9)+5 = 5$	$d>0$	SE	7	7
$5+2(6-8)+5 = 6$	$d>0$	SE	8	6

$x > y$

Bresenham's Mid Point Circle Algorithm

Mathematics (solution): Final Pixels

(x,y)= (1,10)
(2,10)
(3,10)
(4,9)
(5,9)
(6,8)
(7,7)

Bresenham's Mid Point Circle Algorithm

Mathematics (Question): if center is not 0,0

Midpoint circle Example

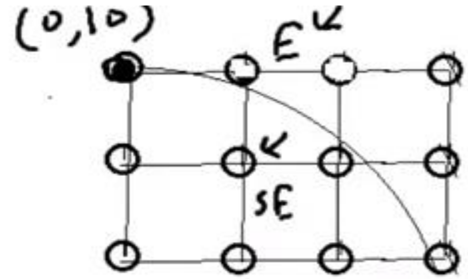
(2,2) → center

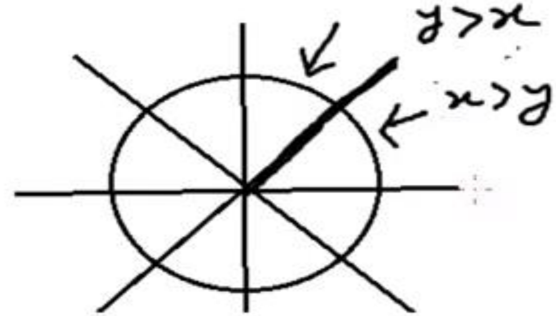
10 — Radius

$d_{start} = 1 - R = 1 - 10 = -9$

$\Delta_E = 2x_p + 3$

$\Delta_{SE} = 2(x_p - y_p) + 5$





Bresenham's Mid Point Circle Algorithm

Mathematics (solution): if center is not zero

d	comparison	Decision	2+0 x	2+10 y
-9	d<0	E	2+1	2+10
-9+2*0+3 = -6	d<0	E	2+2	2+10
-6+2*1+3 = -1	d<0	E	2+3	2+10
-1+2*2+3 = 6	d>0	SE	2+4	2+9

Bresenham's Mid Point Circle Algorithm

Mathematics (solution): if center is not zero

$-6+2*1+3=-1$	$d<0$	E	$2+3$	$2+10$
$-1+2*2+3=6$	$d>0$	SE	$2+4$	$2+9$
$6+2(3-10)+5 = -3$	$d<0$	E	$2+5$	$2+9$
$-3+2*4+3=8$	$d>0$	SE	$2+6$	$2+8$
$8+2(5-9)+5 = 5$	$d>0$	SE	$2+7$	$2+7$

Bresenham's Mid Point Circle Algorithm

Mathematics (solution): if center is not zero

$-3 + 2*4 + 3 = 8$ $\quad d > 0$

$8 + 2(5-9) + 5$
$= 5$ $\quad d > 0$

$5 + 2(6-8) + 5$
$= 6$ $\quad d > 0$

SE $\quad 2+6 \quad 2+8$

SE $\quad 2+7 \quad 2+7$

SE $\quad 2+8 \quad 2+6$

Bresenham's Mid Point Circle Algorithm

Mathematics (solution): if center is not zero

(x,y)= (3,12)
(4,12)
(5,12)
(6,11)
(7,11)
(8,10)
(9,9)