CSEUGPC20-Computer Graphics

UNIT-4

Attribute of Output Primitives

Attribute of Output Primitives

- Definition
- Line Attribute
- Curve Attribute
- COLOR AND GRAY SCALE LEVEL
- AREA FILLED ATTRIBUTE
- Text and Characters

Attributes of Output Primitives

Definition

Parameter that affects the way a primitive will be displayed

Line Attribute

. Type

. Width

. Color

. Pen & Brush

Line Attribute

Type

. Solid _____

. Dotted – very short dash with spacing equal to or greater than dash itself _____

. Dashed – displayed by generating an interdash spacing _____

Pixel count for the span and interspan length is specified by the mask . Ex. 1111000111110001111

Note : Fixed pixel with dashes can produce unequal length dashes. It depend on line orientation. So, need to adjust the number of plotted pixels for different slopes.

Line Attribute

Width

. Specify in pixels and proportion of a standard line width.

. Thicker line can be produced by.

. Adding extra pixel vertically when $|m| < 1$

. Adding extra pixel horizontally when $|m| > 1$

. Issues:

. Line have different thickness on the slope

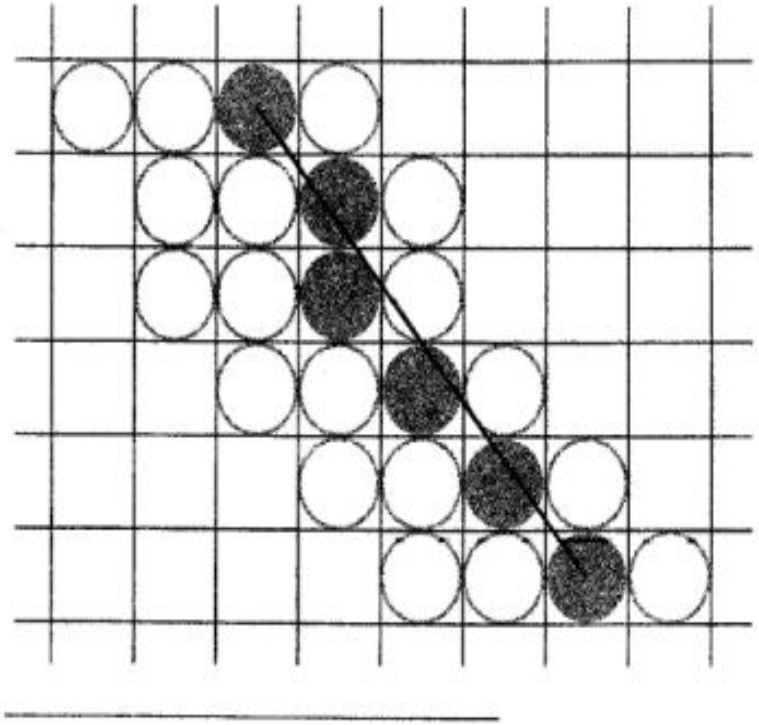
. Problem with

. End of the line

. Joining the two lines (polygon)

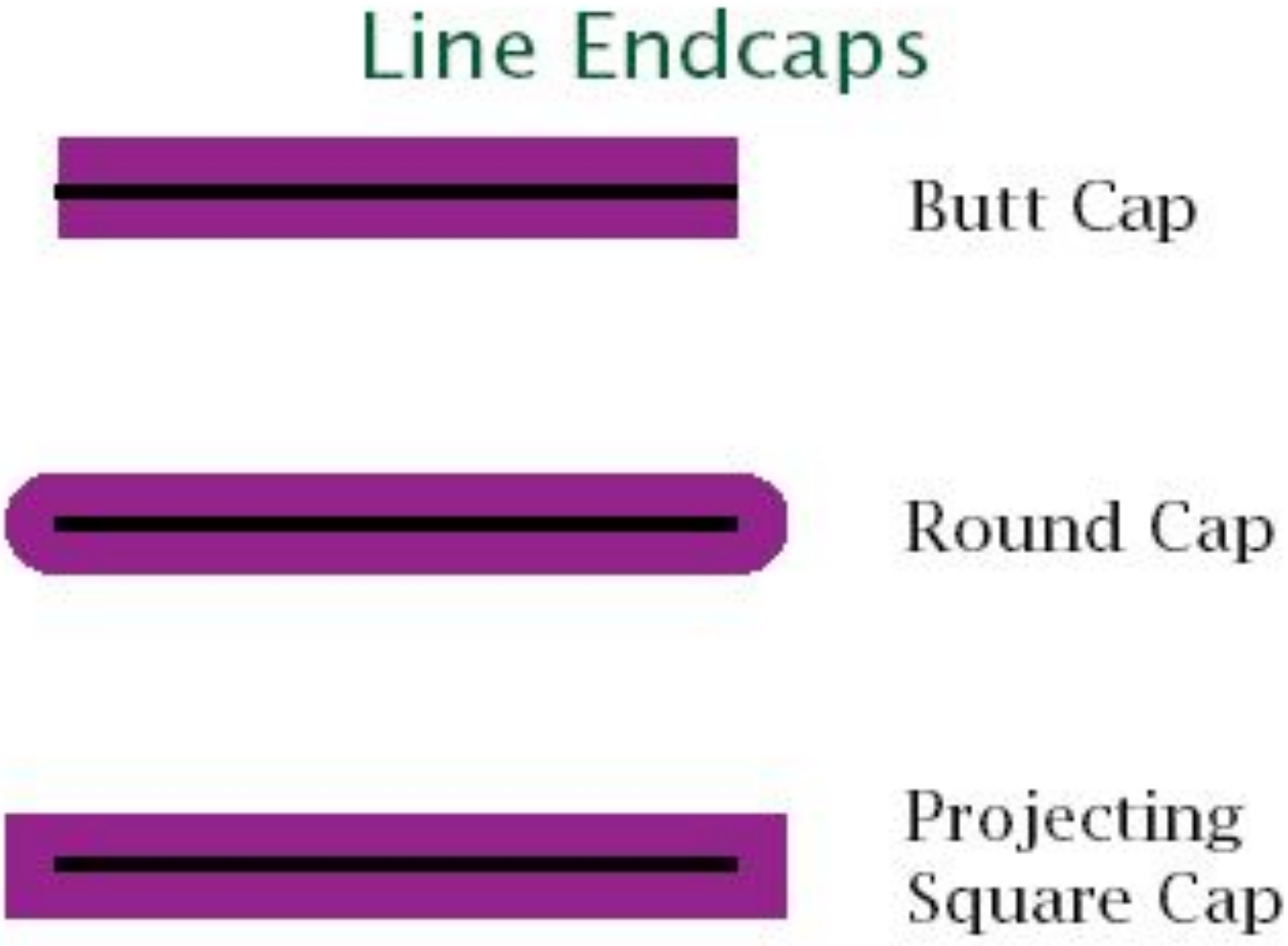
Line Attribute

Width

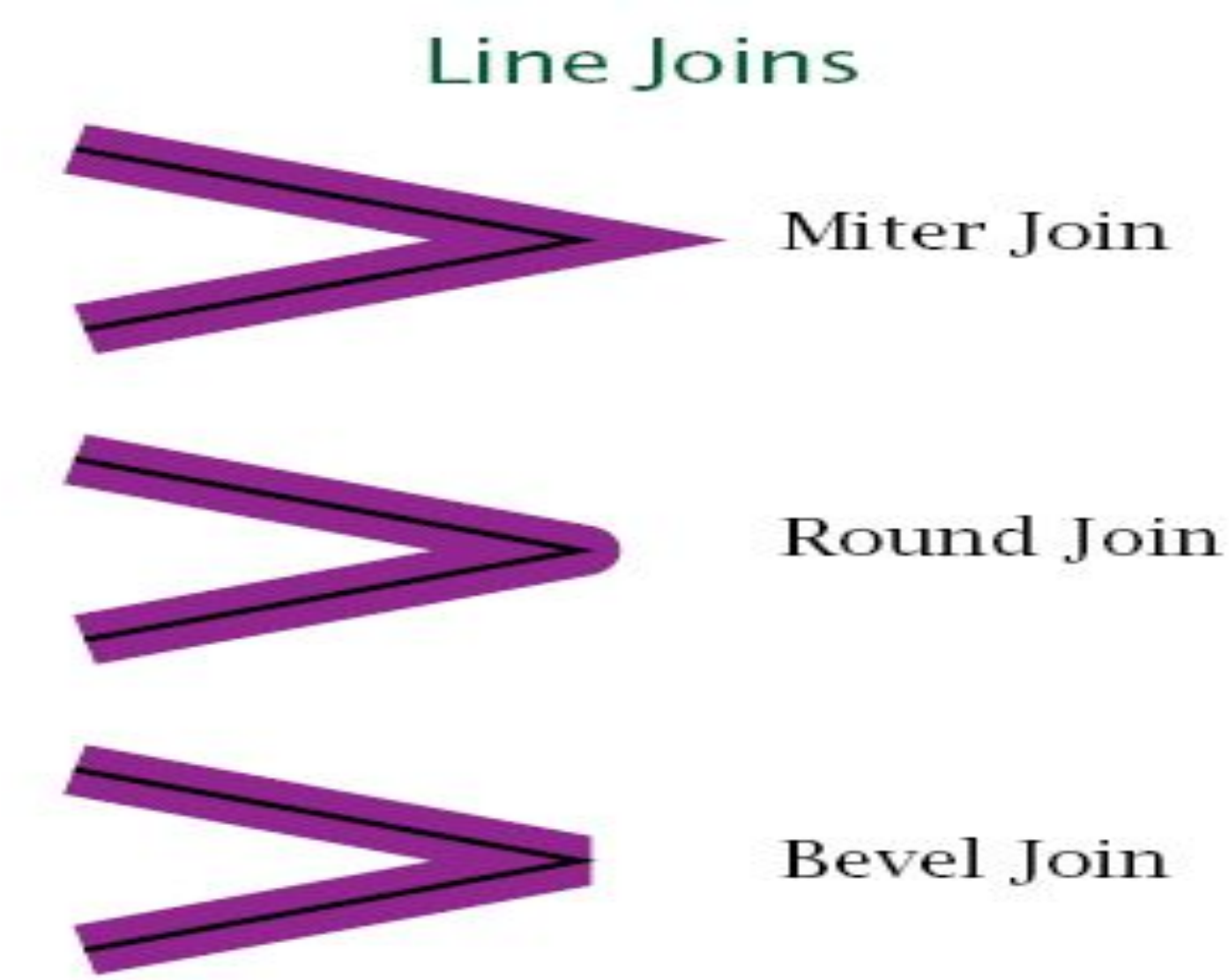


Raster line with slope $|m| > 1$
and line-width parameter `lw` = 4
plotted with horizontal pixel spans.

Attributes of Output Primitives



Attributes of Output Primitives



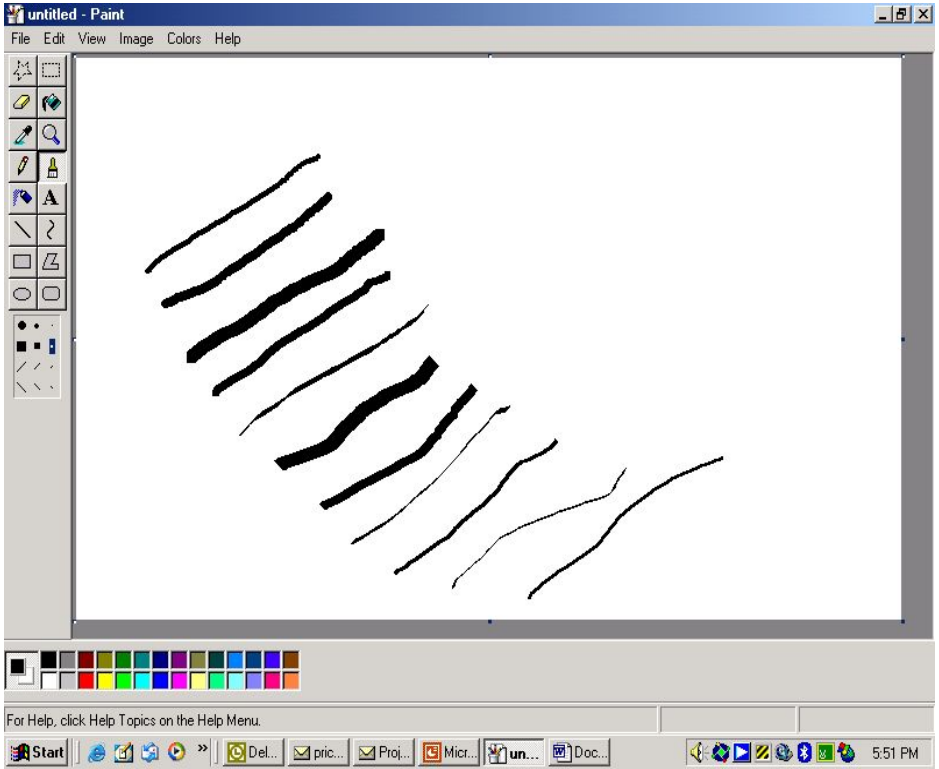
Line Attribute

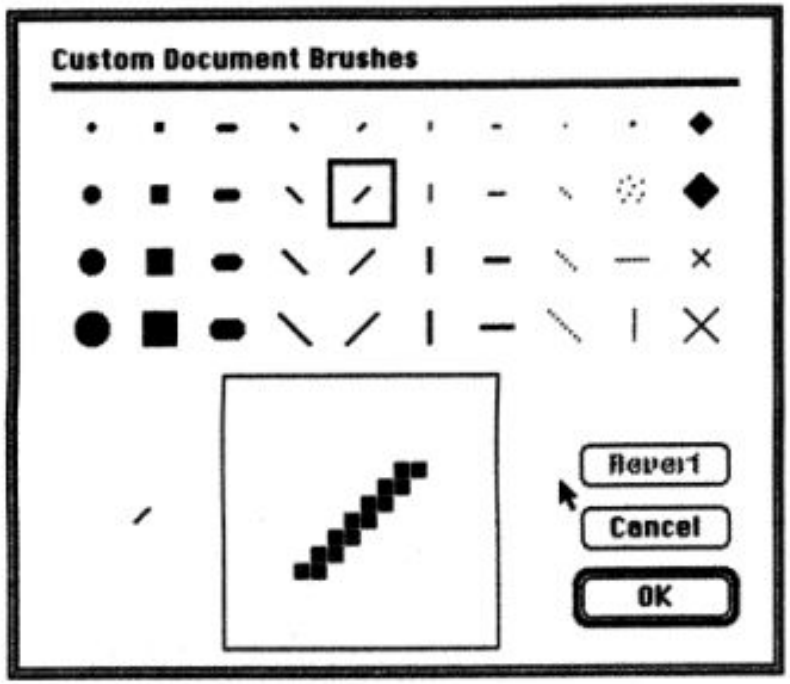
Pen and Brush

. The selected “pen” or “brush” determine the way a line will be drawn.

. Pens and brushes have size, shape, color and pattern attribute.

. Pixel mask is applied in both of them.



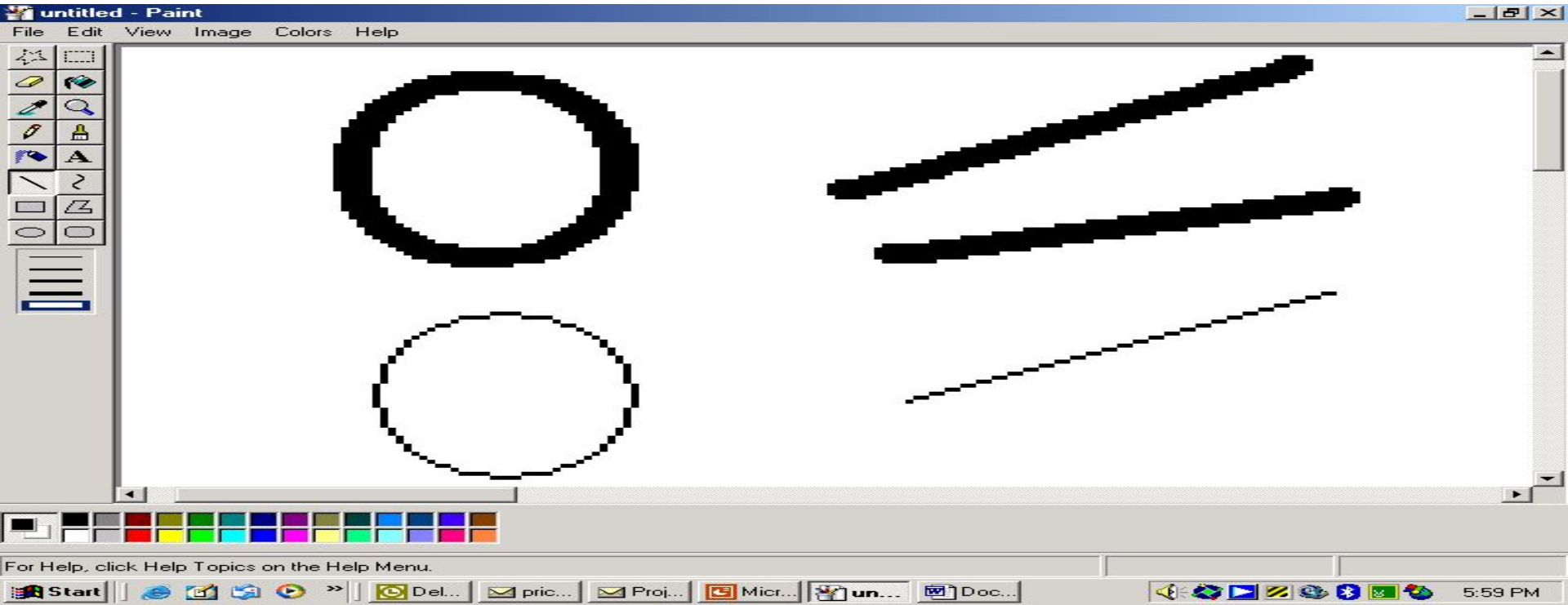


Curve Attribute

Similar to line : type + width

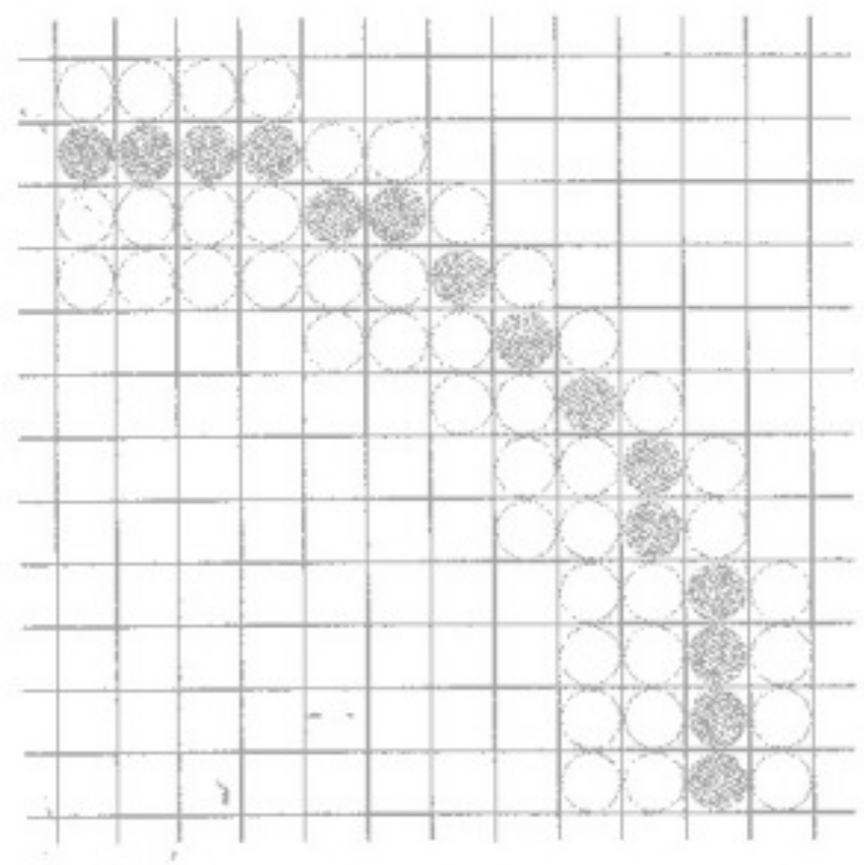
Thicker curves can be produced by:

1. Plotting additional pixel
2. Filling the space between two concentric circles.
3. Using thicker pen or brush



Curve Attribute

Width



Circular arc of width 4 plotted with pixel spans.

COLOR AND GRAY SCALE LEVEL

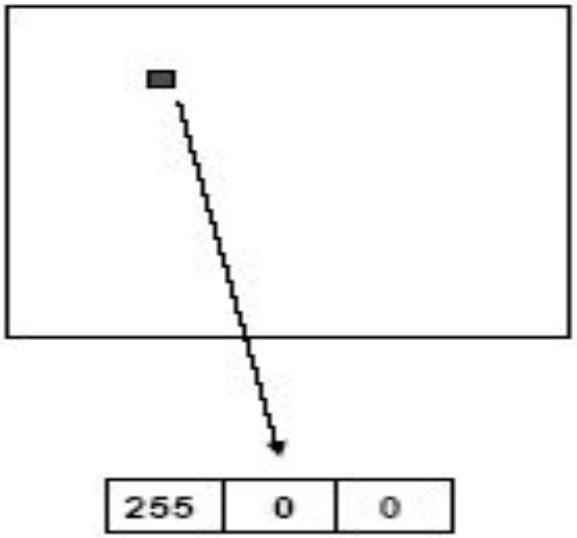
Color

- Colors are represented by colors codes which are positive integers.
- Color information is stored in frame buffer or in separate table and use pixel values as index to the color table.
- Two ways to store color information :
  1. Direct
  2. Indirect

COLOR

Direct

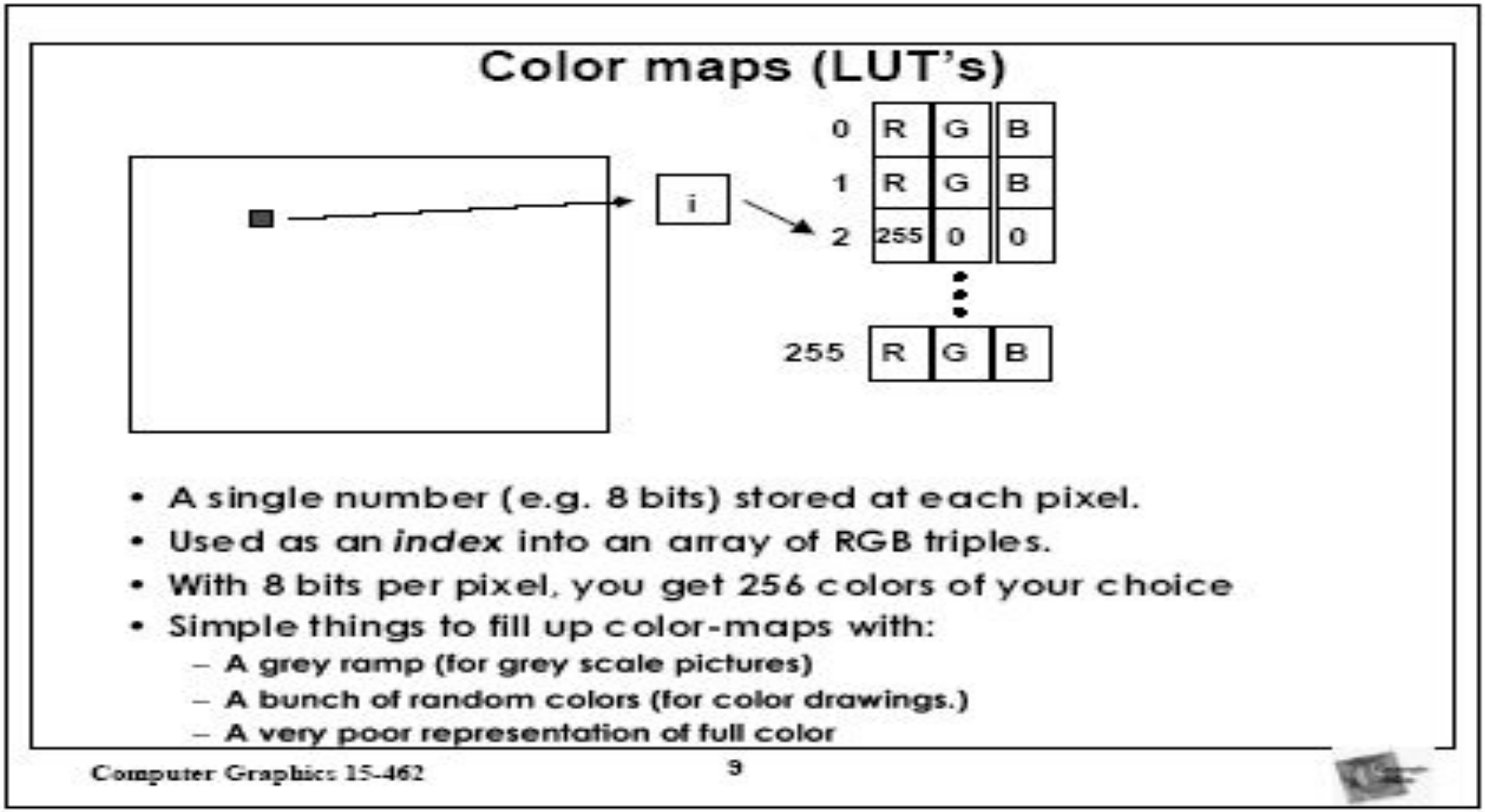
Full-color (RGB) displays



- For 24 bit color:
  - store 8 bits each of red, green, and blue per pixel.
  - E.g. (255,0,0) is pure red, and (255, 255, 255) is white.
  - Yields 2^24 = 16 million colors.
- For 15 bit color:
  - 5 bits red + 5 green + 5 blue
- The video hardware uses the values to drive the R,G, and B guns.
- You can mix different levels of R, G, and B to get any color you want

COLOR

Indirect



COLOR

Exercise:

What is the size of frame buffer required for the following cases:

Case 1 (Direct): Resolution of 1024 * 1024 with 24 bits per pixel

Case 2 (Indirect): Same resolution with 8 bit per pixel that indexed out of 16 million available colors

Conclusion

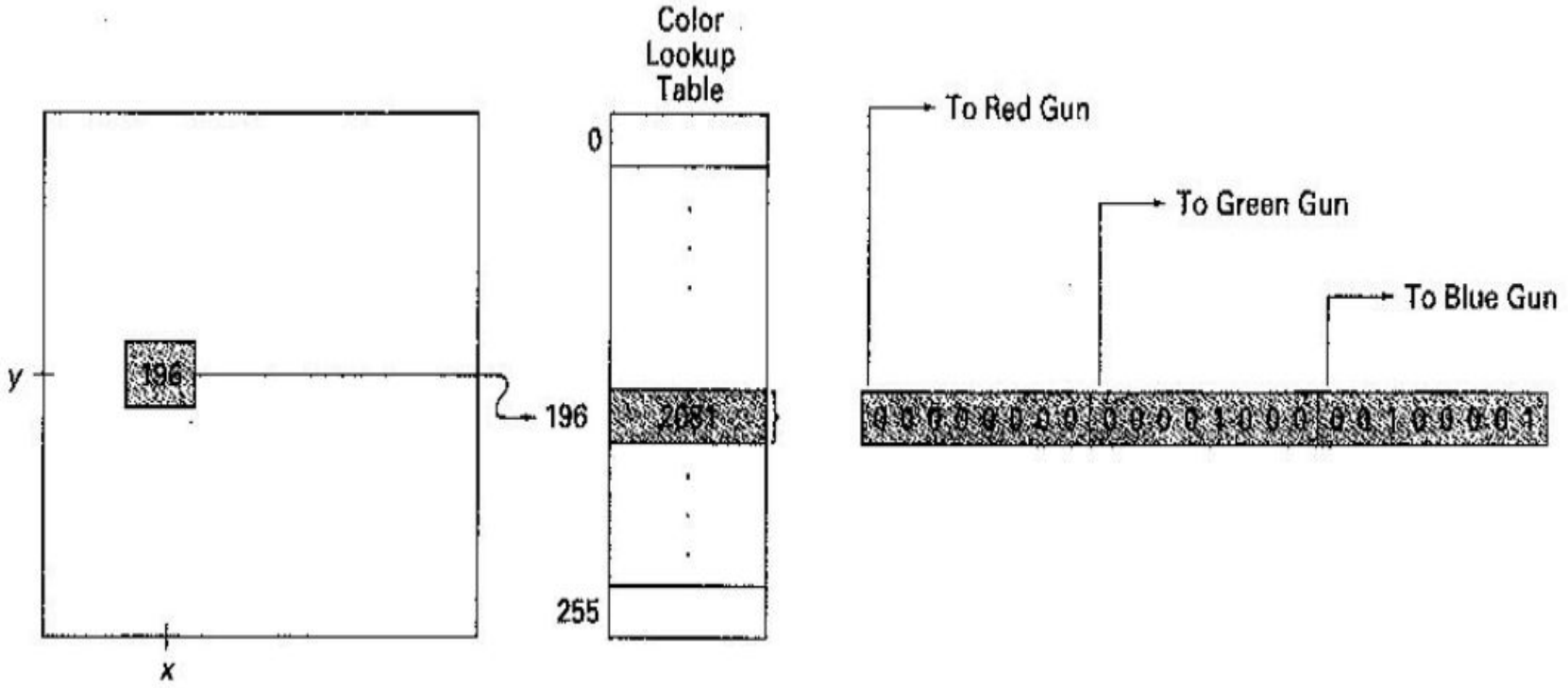
CLUT is good for storage but cant give a very high resolution picture.

COLOR Lookup Table

THE EIGHT COLOR CODES FOR A THREE-BIT PER PIXEL FRAME BUFFER

Color	*Stored Color Values in Frame Buffer*			*Displayed Color*
Code	*RED*	*GREEN*	*BLUE*	
0	0	0	0	Black
1	0	0	1	Blue
2	0	1	0	Green
3	0	1	1	Cyan
4	1	0	0	Red
5	1	0	1	Magenta
6	1	1	0	Yellow
7	1	1	1	White

COLOR Lookup Table



A color lookup table with 24 bits per entry accessed from a frame buffer with 8 bits per pixel. A value of 196 stored at pixel position (x, y) references the location in this table containing the value 2081. Each 8-bit segment of this entry controls the intensity level of one of the three electron guns in an RGB monitor.

GRAY SCALE LEVEL

.Apply for monitor that have no color
.Shades of grey (white->light grey->dark grey->black)
.Color code mapped onto grayscale codes
.2 bits can give 4 level of grayscale
.8 bits per pixel will allow 256 combination
.Dividing the actual code with 256 will give range of 0 and 1

Ex:

Color code in color display is 118

To map to nearest grayscale then

118/256 = 0.45

 light gray

INTENSITY CODES FOR A FOUR-LEVEL GRAYSCALE SYSTEM

Intensity Codes	*Stored Intensity Values In The Frame Buffer (Binary Code)*		*Displayed Grayscale*
0.0	0	(00)	Black
0.33	1	(01)	Dark gray
0.67	2	(10)	Light gray
1.0	3	(11)	White

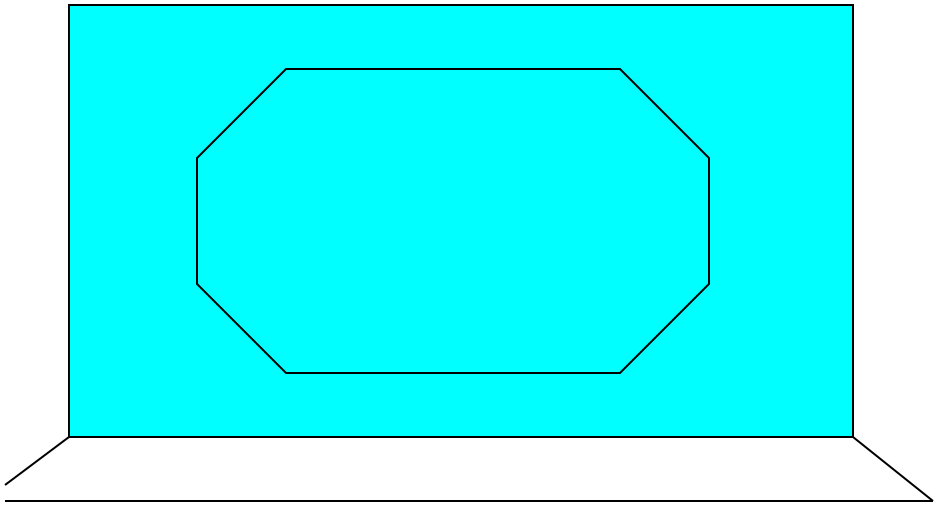
AREA FILLED ATTRIBUTE

. Option for filling a defined region is whether solid , pattern and colors.

Fill Styles

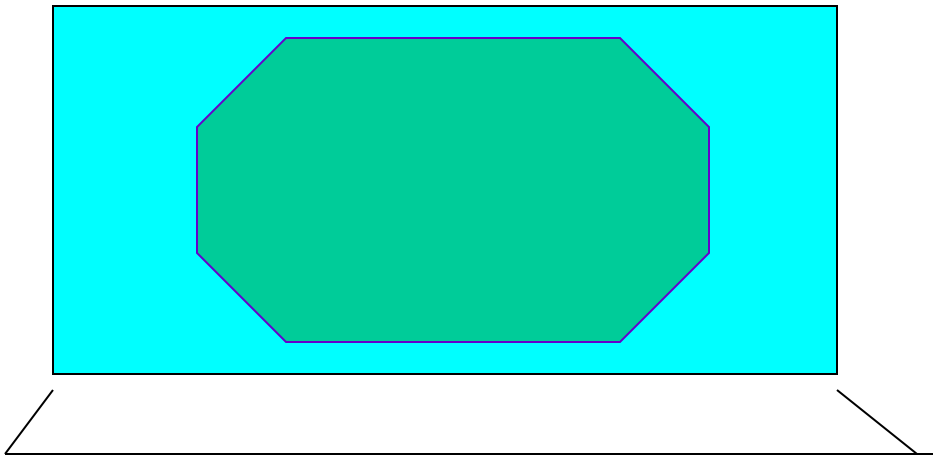
. Three basic fill styles are:

. hollow with color border.. interior color is same with background

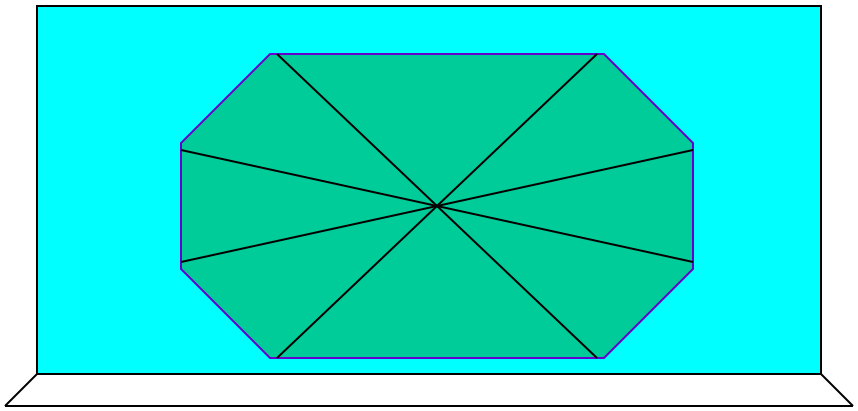


AREA FILLED ATTRIBUTE

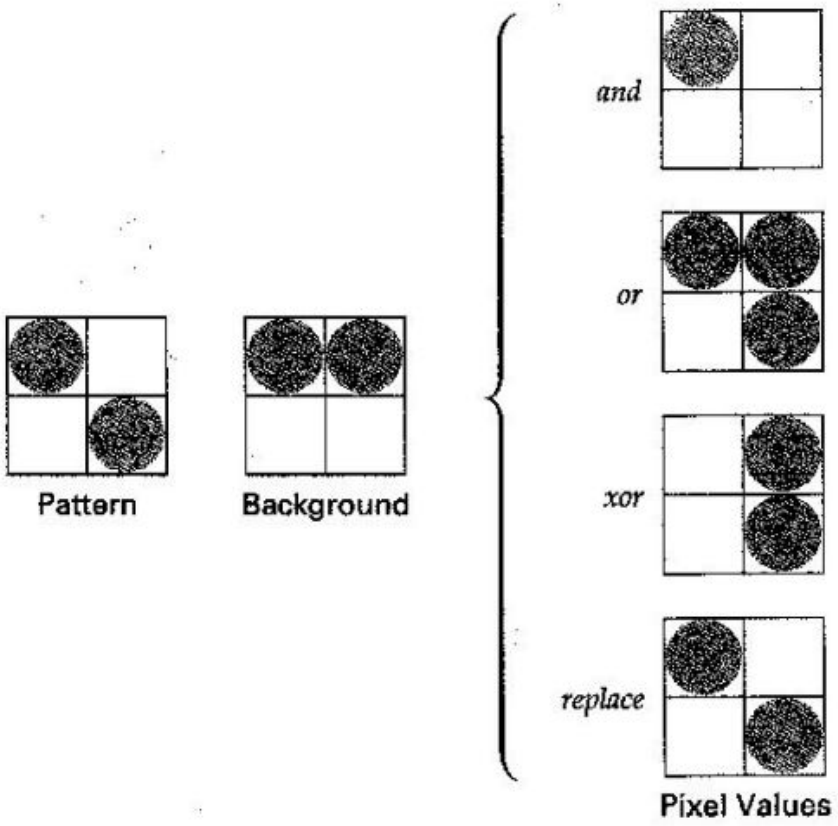
. filled with a solid color .. color up to and including the border



pattern .. control by other table



AREA FILLED ATTRIBUTE



Combining a fill pattern with a background pattern using Boolean operations, *and*, *or*, and *xor* (*exclusive or*), and using simple replacement.

AREA FILLED ATTRIBUTE

. Color-Blended Fill Region

-Soft-fill

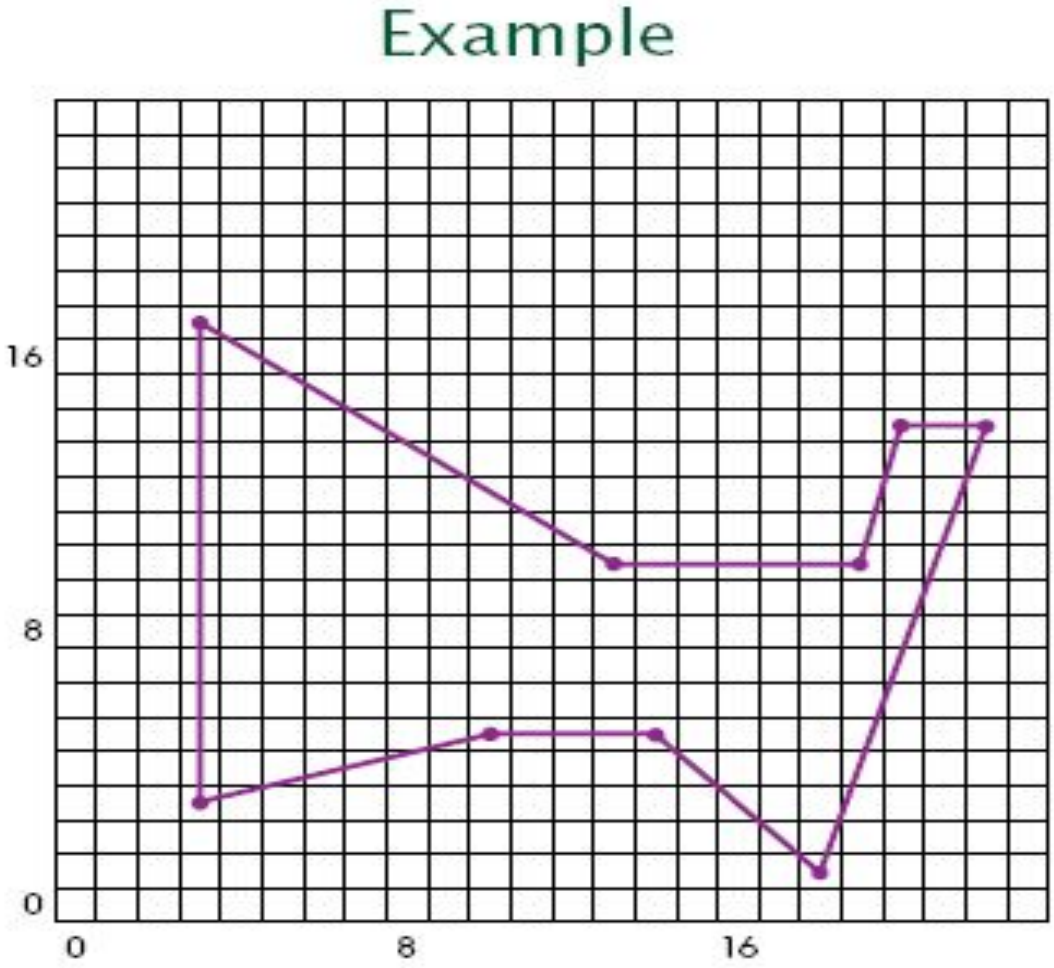
- $P = tF + (1-t)B$ where $0 < t < 1$

If $t < 0.5$, background color contributes more to the interior color of the region than does the fill color.

RGB component of the colors, with

$$\mathbf{P} = (P_R, P_G, P_B), \qquad \mathbf{F} = (F_R, F_G, F_B), \qquad \mathbf{B} = (B_R, B_G, B_B)$$

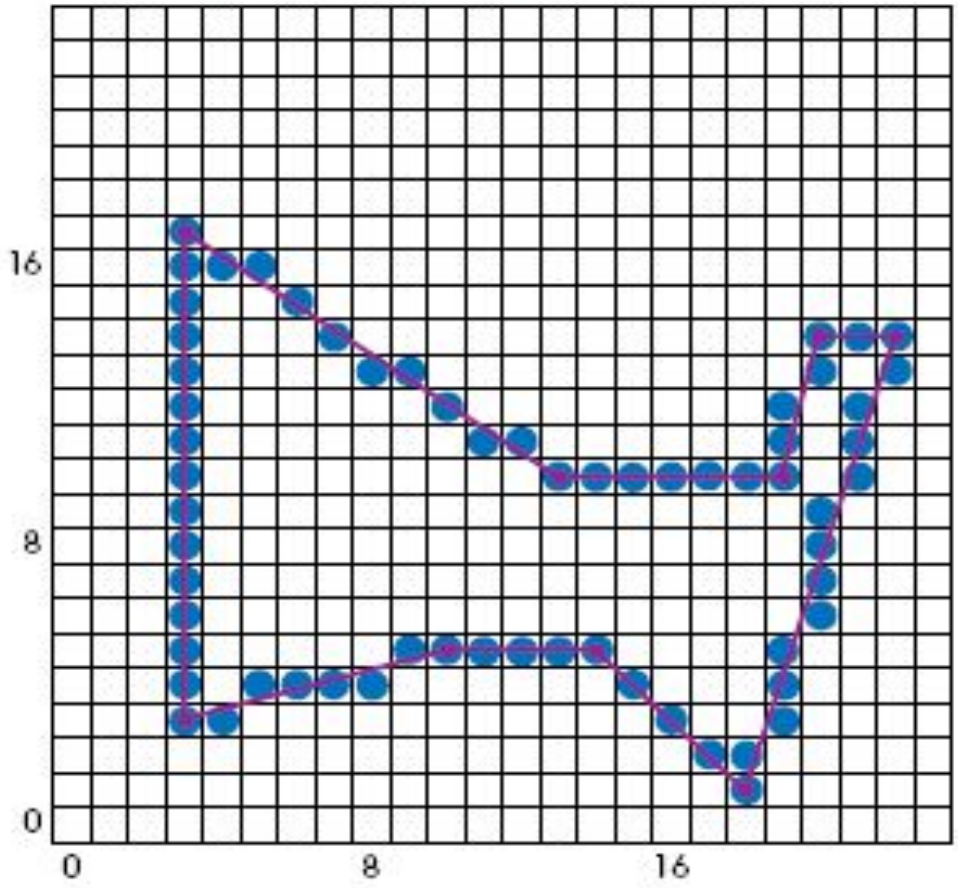
Filled-Area Primitives



The boundary of a polygon: (In practice, a polygon is defined as a list of vertices.)

Filled-Area Primitives

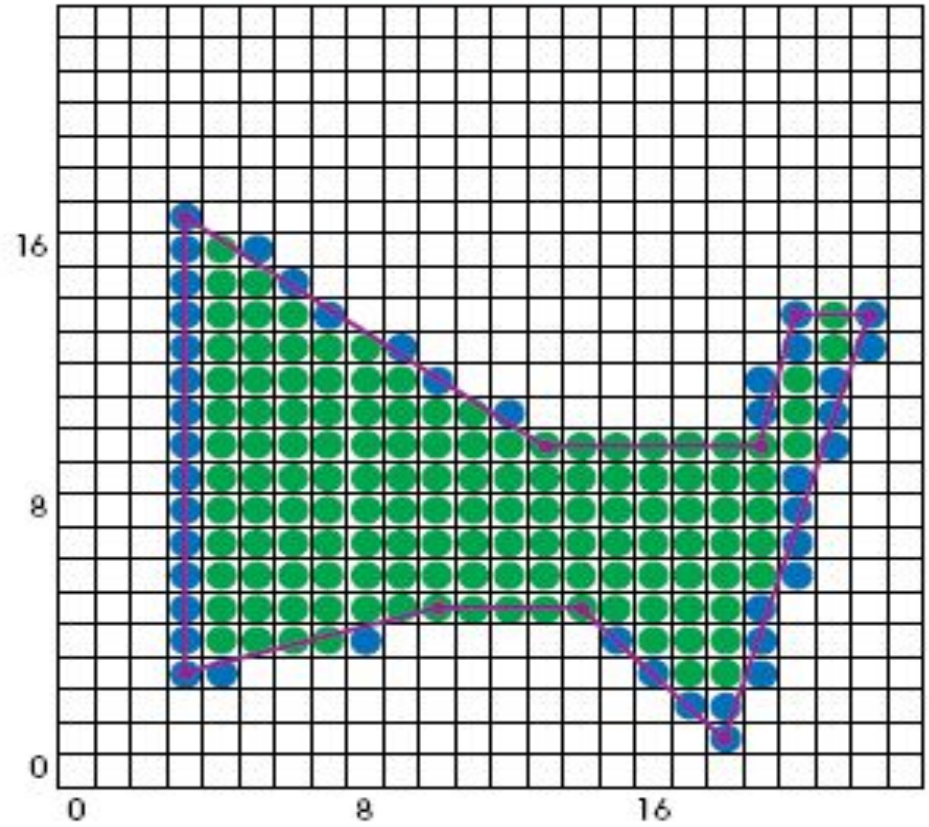
Example (cont.)



The pixels that are closest to the boundary of a polygon, obtained from Bresenham's algorithm, for instance.

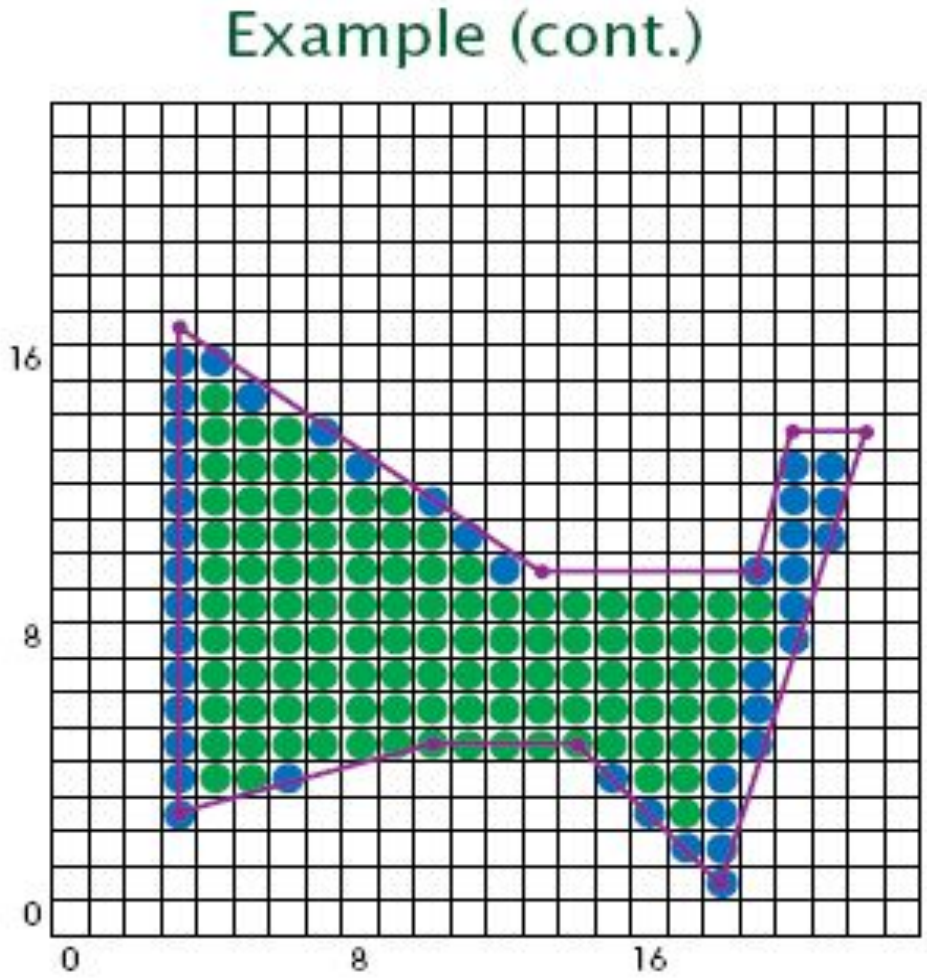
Filled-Area Primitives

Example (cont.)



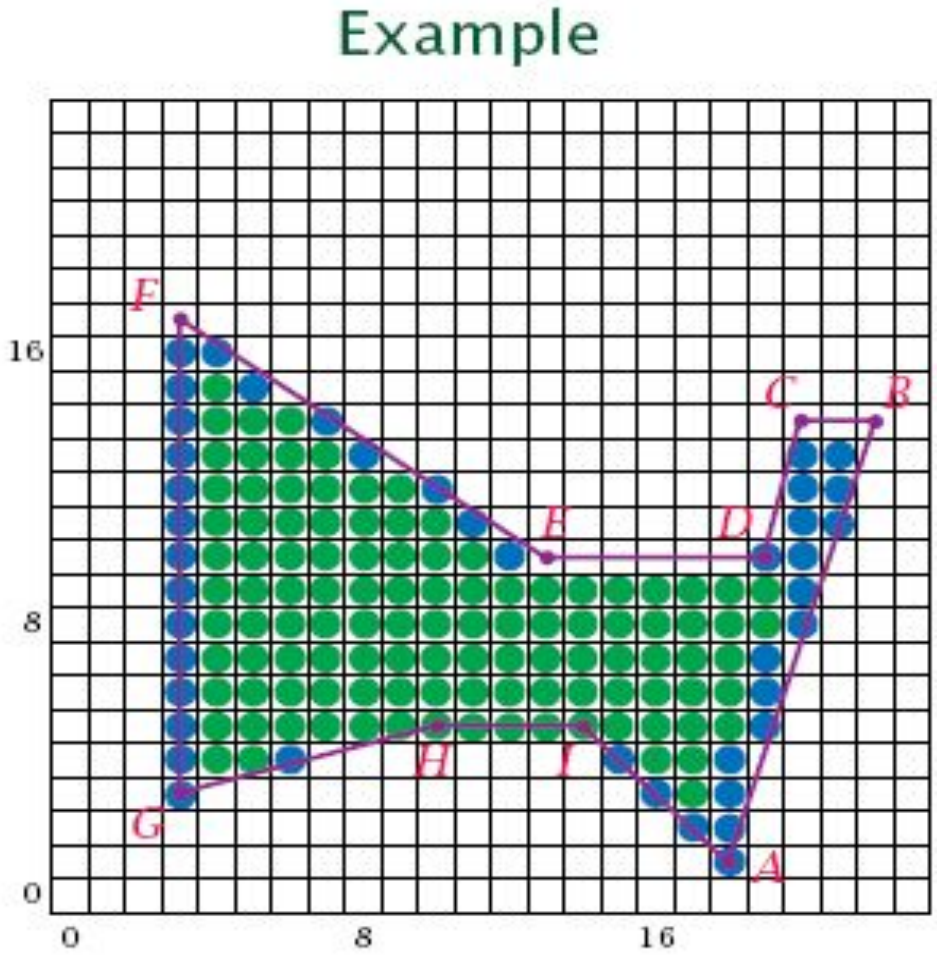
The scan extrema (blue) and interior (green) pixels that would be obtained if the span extrema were defined by the pixels closest to the boundary. Our convention excludes some of these.

Filled-Area Primitives



The scan extrema (blue) and interior (green) pixels that are obtained using our interior pixel convention for the given polygon (purple).

Filled-Area Primitives



The scan extrema (blue) and interior (green) pixels that are obtained using our interior pixel convention for the given polygon (purple).

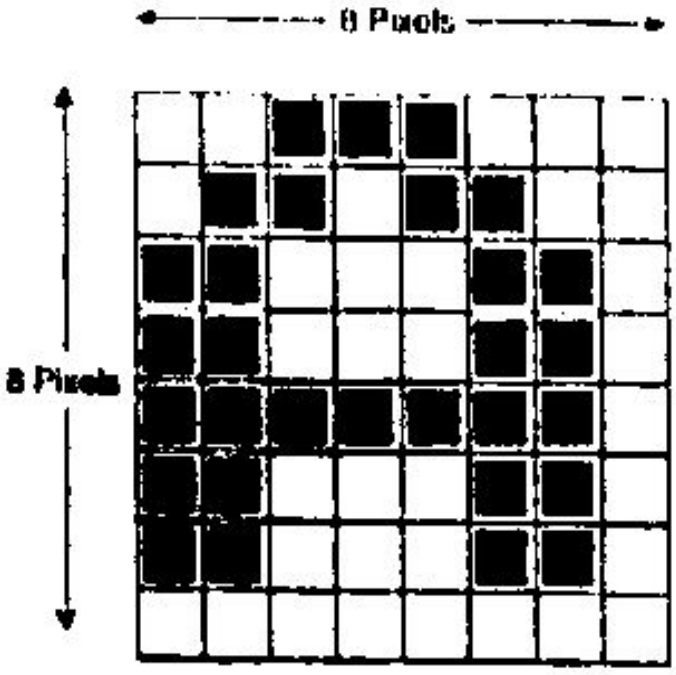
Text and Characters

Text and Characters

- Very important output primitive
- Many pictures require text
- Two general techniques used
  - Bitmapped (raster)
  - Stroked (outline)

Text and Characters (Bitmapped (raster))

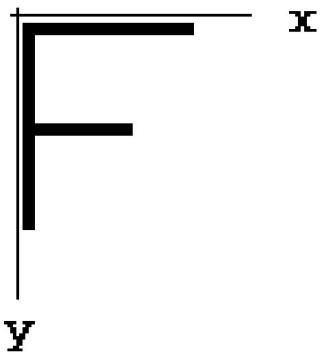
Each character represented (stored) as a 2-D array
– Each element corresponds to a pixel in a rectangular "character cell"
– Simplest: each element is a bit (1=pixel on, 0=pixel off)



00111000
01101100
11000110
11000110
11111110
11000110
11000110
00000000

Text and Characters (Stroked (outline))

Each character represented (stored) as
a series of line segments
– sometimes as more complex primitives
Parameters needed to draw each stroke
– endpoint coordinates for line segments



Strokes:

(0,0), (0,10)
(0,0), (10,0)
(0,5), (6,5)

Characters Characteristics

Characteristics of Bitmapped Characters

- Each character in set requires same amount of memory to store
- Characters can only be scaled by integer scaling factors
- "Blocky" appearance
- Difficult to rotate characters by arbitrary angles
- Fast (BitBLT)

Characteristics of Stroked Characters

- Number of stokes (storage space) depends on complexity of character
- Each stroke must be scan converted more time to display
- Easily scaled and rotated arbitrarily
- – just transform each stroke

Bundled Attributes

When each function reference a single attribute that specify exactly how primitive to be displayed ; then its called ***individual (unbundled attribute).***

.***Bundled*** attributes is where a set of attributes value can be chosen by specifying the appropriate index table.

.The table for each primitive that defines group of attribute values is called ***bundled table.***

.Attribute that can be bundled are:

- . Bundled Line Attribute
- . Bundled Area-Fill Attribute
- . Bundled Text Attribute
- . Bundled Marker Attribute