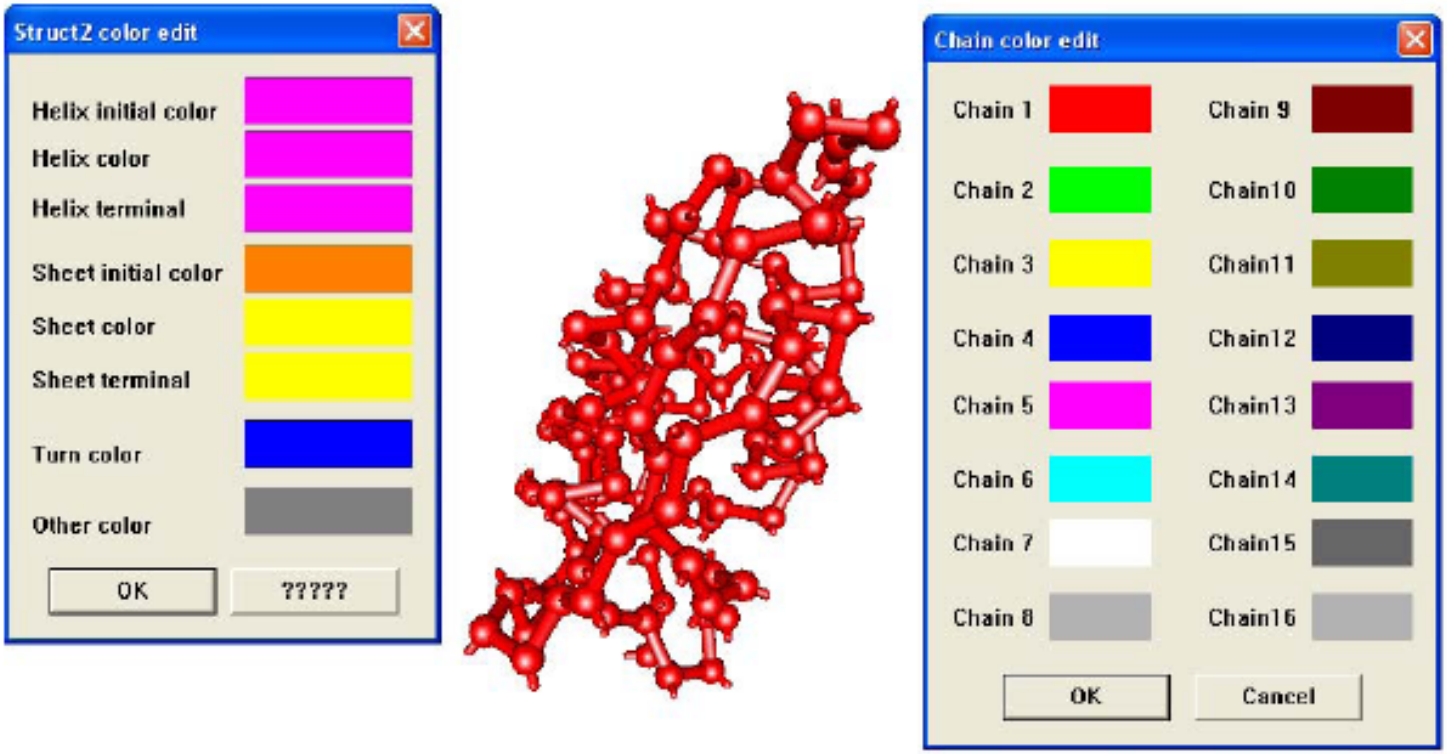
Chapter 5: Attributes

Objective

- To study attributes in computer graphics included: line style/attributes, color & intensity, and character attributes.
- Introduction to graphics system using OpenGL.

Introduction

Any parameters that affects the way a primitive to be displayed is as attribute primitive.



Introduction

- How primitives are to be displayed
- Most systems use modal attributes
  - Values in effect until changed

Line Attributes

- Color
- Width
- Style--solid, dotted, dashed, etc.

  Can be specified by giving a pattern array

  e.g., pat[ ]={1,1,1,1,1,1,0,0}

  Repeat this pattern on entire line:

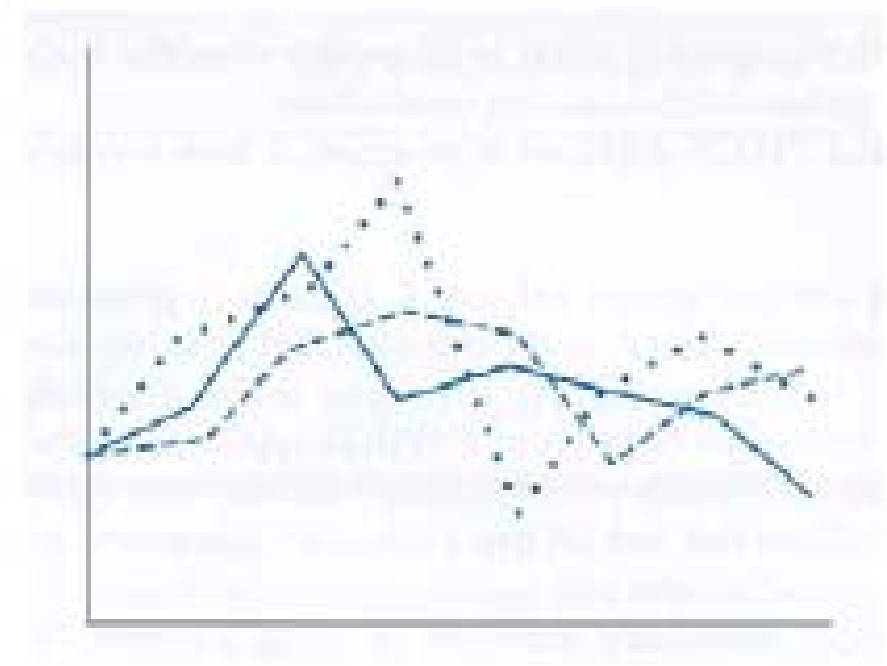
  i^{th} pixel along line:

  ```
  if (pat[i%8]==1) SetPixel(x,y)
  ```

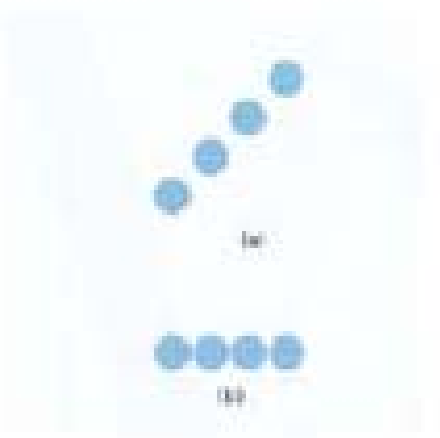
- In Windows, use a pen (CPen)

Line Attributes (cont'd)

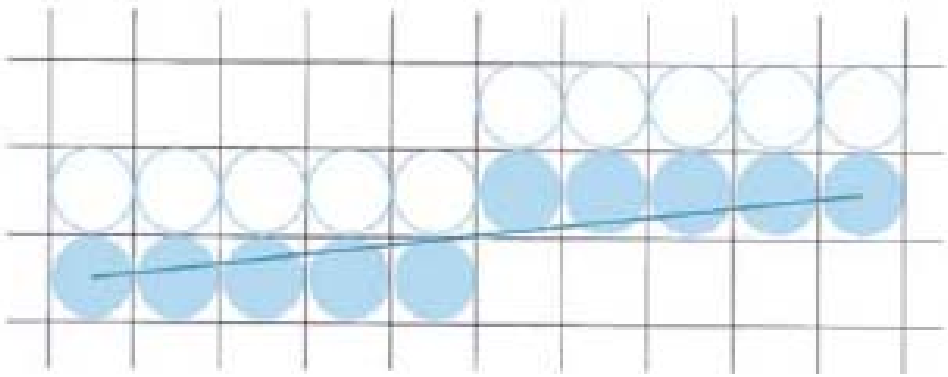
- *Type:* solid dotted
  - a very short dash with spacing equal to or greater than dash itself.
  - Dashed : displayed by generating an inter-dash spacing.
- Pixel count for the span and inter-span length is specified in the mask.
  - Ex: 1111000111110001111
- Fixel pixel width dashes can produce unequal length dashes.Its depend on line orientations.
- So, need to adjust the number of pixels plotted for different slopes.



Line Type



unequal length



Width

- *Width*

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

  .Thicker line can be produced by.

  . adding extra pixel horizontally when m < 1

  . adding pixel vertically when m > 1

Issues:

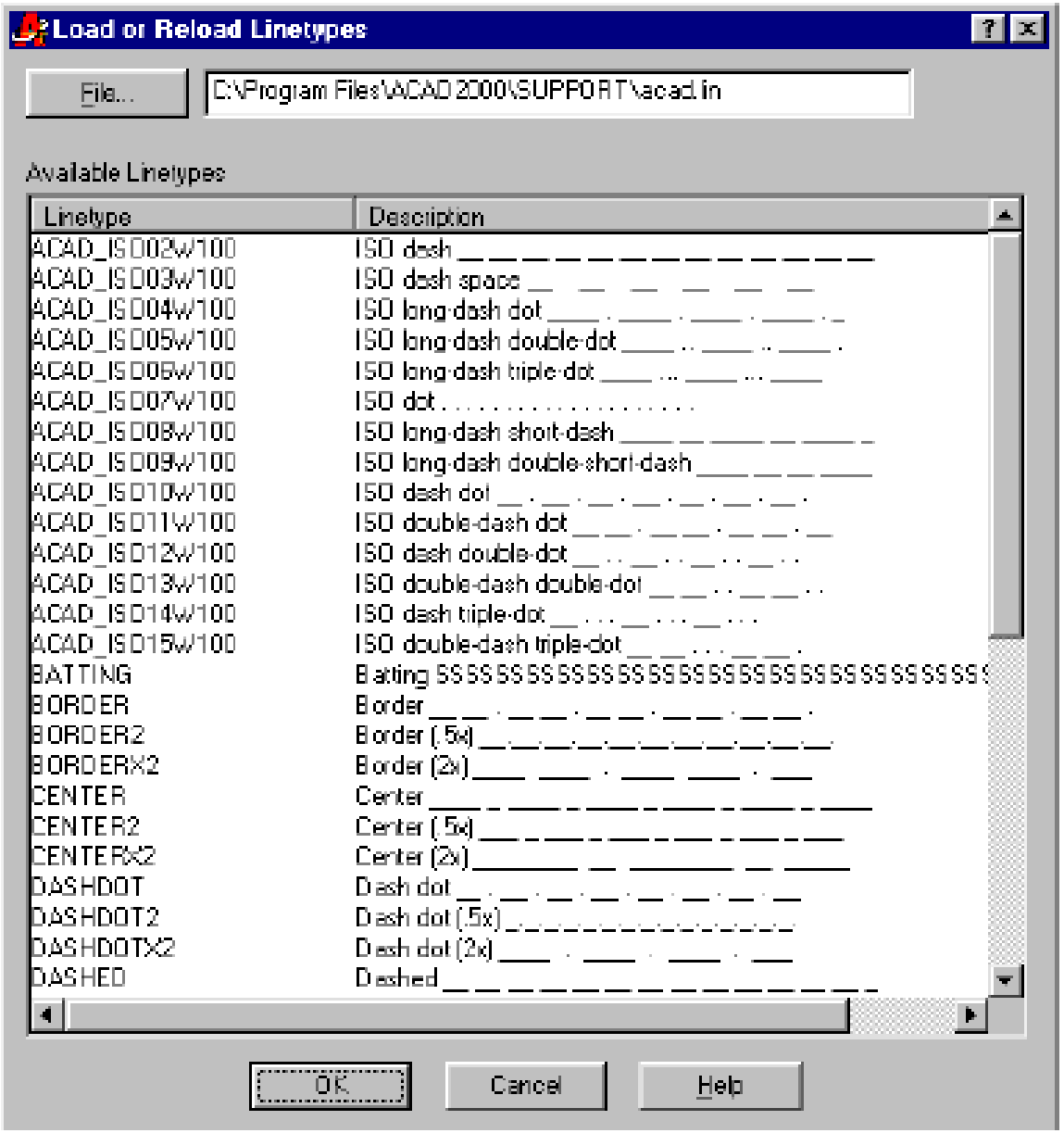
- Lines have different thickness on the slope.
- To have nice end on the line: butt cap, round cap, projecting square cap are introduced.
- For solving joining problems in polygon. This method is for producing smooth connection when shifting from horizontal spans to vertical spans.

  . mitre joins
  . round joins
  . bevel join.

- *Pen and Brush options*
  - The selected "pen" or "brush" determine the way a line will be drawn
  - Pens and brushes have size,shape,color and pattern attributes.
  - Pixel mask is applied in pens and brushes.

Curve Attributes

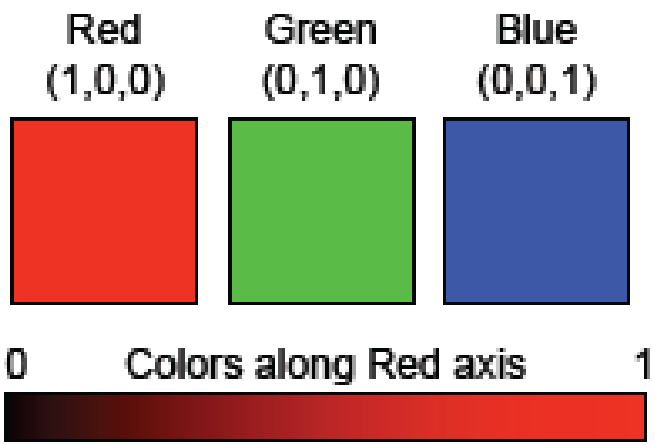
- Same attributes as lines.
- Thicker curves can be produces by
  - Plotting additional pixels.
  - Filling the space between two concentric circles. However will cause inward or outward depend on second boundary.
  - Using thicker pen or brush.

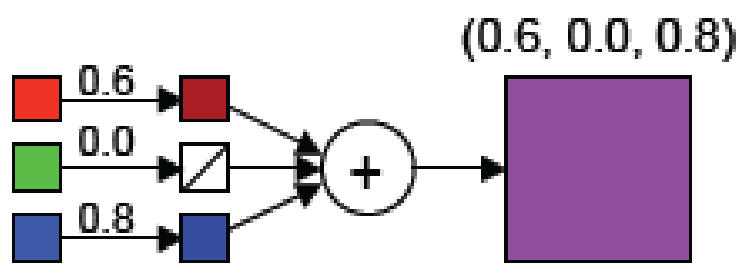

Load or Reload Linetypes
File...
C:\Program Files\ACAD 2000\SUPPORT\acad.lin
Available Linetypes
Linetype
Description
ACAD_ISO02W100
ISO dash
ACAD_ISO03W100
ISO dash space
ACAD_ISO04W100
ISO long-dash dot
ACAD_ISO05W100
ISO long-dash double-dot
ACAD_ISO06W100
ISO long-dash triple-dot
ACAD_ISO07W100
ISO dot
ACAD_ISO08W100
ISO long-dash short-dash
ACAD_ISO09W100
ISO long-dash double-short-dash
ACAD_ISO10W100
ISO dash dot
ACAD_ISO11W100
ISO double-dash dot
ACAD_ISO12W100
ISO dash double-dot
ACAD_ISO13W100
ISO double-dash double-dot
ACAD_ISO14W100
ISO dash triple-dot
ACAD_ISO15W100
ISO double-dash triple-dot
BATTING
Batting
BORDER
Border
BORDER2
Border (.5x)
BORDERX2
Border (2x)
CENTER
Center
CENTER2
Center (.5x)
CENTERX2
Center (2x)
DASHDOT
Dash dot
DASHDOT2
Dash dot (.5x)
DASHDOTX2
Dash dot (2x)
DASHED
Dashed
OK
Cancel
Help

Basic Color Representation in Graphics

We will treat colors as a 3-D space of (r, g, b) triples

- all colors will be composed from three primary colors: red, green, blue
- the value of each (r, g, b) value is between 0 and 1
- coefficients represent relative contribution of each primary





Full-Color Display

Each pixel contains 3 values, one for each of R, G, and B

- typically 24 bits/pixel = 8 bits/channel = values of 0–255
- integer values 0–255 correspond to floating points values 0–1
- integers are just more convenient in hardware implementation

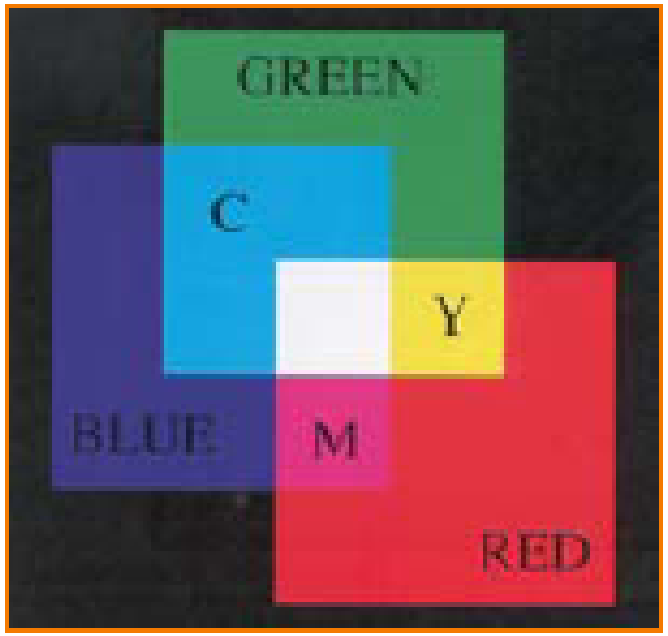
Pixel values directly control intensity of electron beams

- R=0 implies red beam is off
- R=255 implies red beam at full intensity

24 bits/pixel generally considered "full-color"

- produces $2^{24} \approx 16$ million different colors
- high-end systems might support 36 bits/pixel or more

RGB Color Model

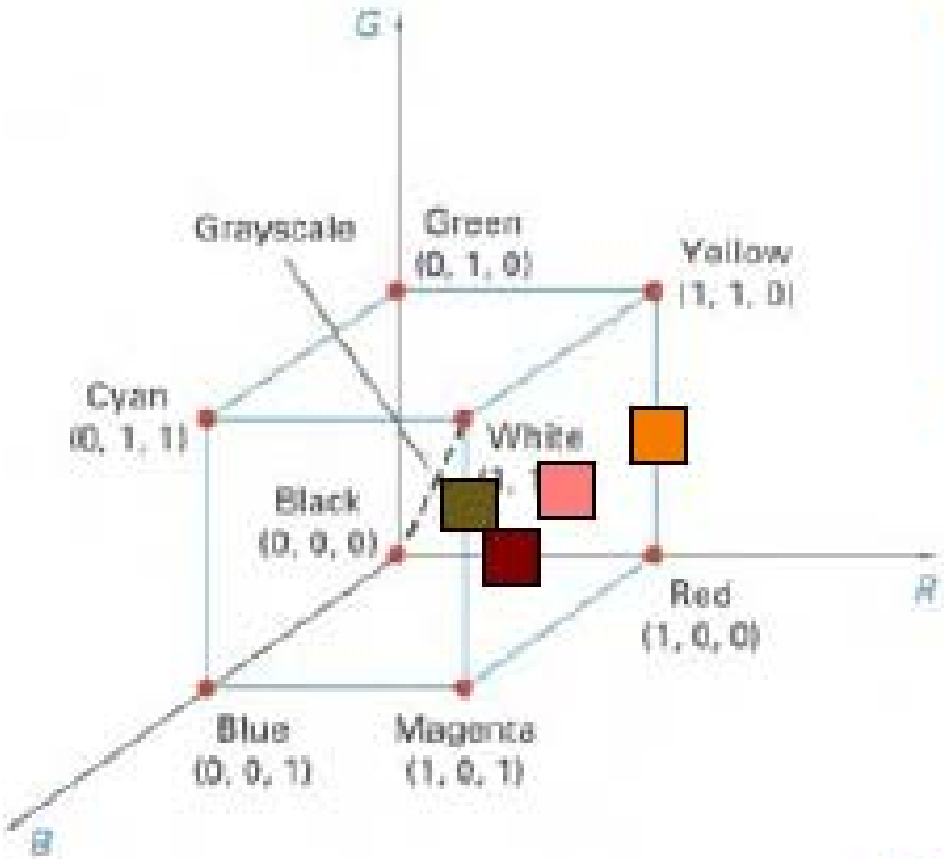


Colors are additive

R	G	B	Color
0.0	0.0	0.0	Black
1.0	0.0	0.0	Red
0.0	1.0	0.0	Green
0.0	0.0	1.0	Blue
1.0	1.0	0.0	Yellow
1.0	0.0	1.0	Magenta
0.0	1.0	1.0	Cyan
1.0	1.0	1.0	White
0.5	0.0	0.0	?
1.0	0.5	0.5	?
1.0	0.5	0.0	?
0.5	0.3	0.1	?

Plate II.3 from FvDFH

RGB Color Cube

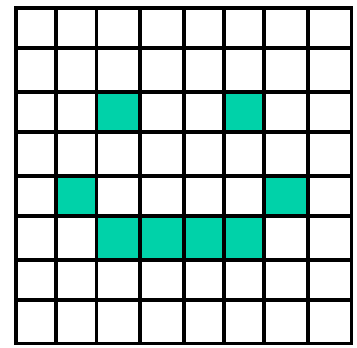


Figures 15.11&15.12 from H&B

Raster Image Representation

Represent image as a rectangular grid of pixels $P[x,y]$

- each pixel p will store a color value
  - RGB triple for color images
  - single value for grayscale (or monochrome) images



Raster Image Representation

Can separate RGB color image into 3 distinct color channels

- each by itself is a monochrome image

Color & Intensity

The main attributes that related to color are:

- COLOR
- GRAYSCALE LEVELS
- Black-White

Color

- Colors are represented by color codes which are positive integers.
- Color information is stored in frame buffer or in a separate table and use pixel values as index to the color table.
- RGB concept (R,G,B): Example is (255,50,100);
- Color codes used to:
  - Control the intensity of of the electron in CRT monitors.
  - Control the choice of pen in plotter, etc.
- Number of color depends on the number of bits per pixel allowed by the size of frame buffer.

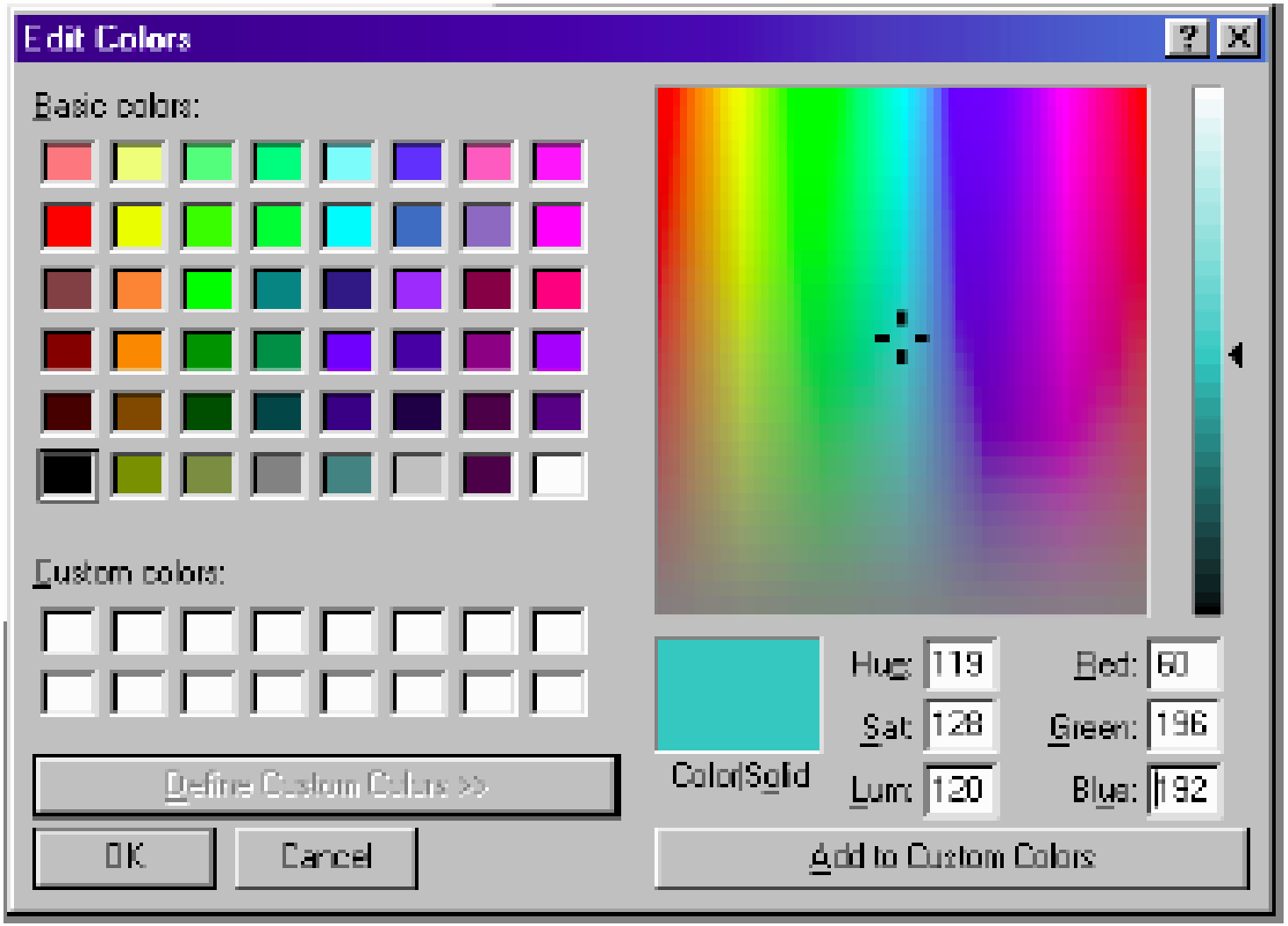
How to Save the Color

- Directly
- Indirectly

- Directly
  - Each pixel contains the numeric color code.
  - 24 bits per pixel allow nearly 17 million different color to be displayed simultaneously.
  - For a resolution of 1024 * 1024 with 24 bits required 3 MB.

- Indirectly
  - The color codes are stored in a separate ***Color Look-Up Table CLUT,*** Each pixel contains an index to the color code.
  - Reduce the number of simultaneous color – need to choose from a palette – but the frame buffer size is also reduced.
  - 8 bits per pixel allow 256 colors to be indexed in the CLUT.
  - The CLUT uses 24 bits to represent each color.
  - So, only 256 colors can be indexed out of 17 million color in the palette.
  - For this example: for a resolution of 1024 * 1024 with 8 bits then need 1 MB.
  - The CLUT requireds 256 * 24 = 6144 bits = 0.75KB
  - The conclusion is that CLUT 'Indirectly' is good for less storage but cant give a very high resolution picture.

RGB with Paintbrush

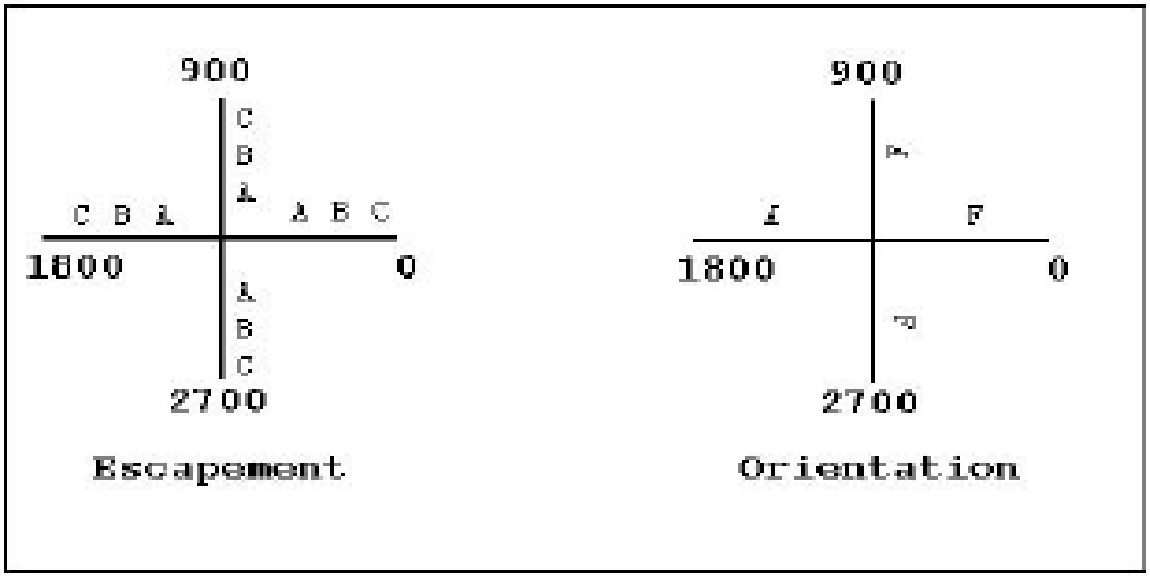


Text Attributes

- Font (typeface)
  - Character set with particular design style
- Display style
  - underlined, italic, boldface, outlined, strikeout, spacing, etc.
- Color
- Size (width, height)--specified in points
  - Point = 1/72 inch

Text Attributes, continued

- Orientation--how much character is rotated
- Escapement--orientation of line between first & last character in a string



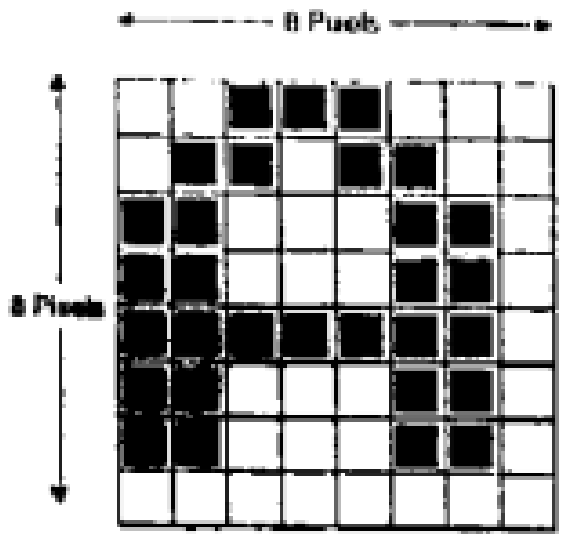
Character Escapement & Orientation

Text and Characters

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

Bitmapped Characters

- 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)



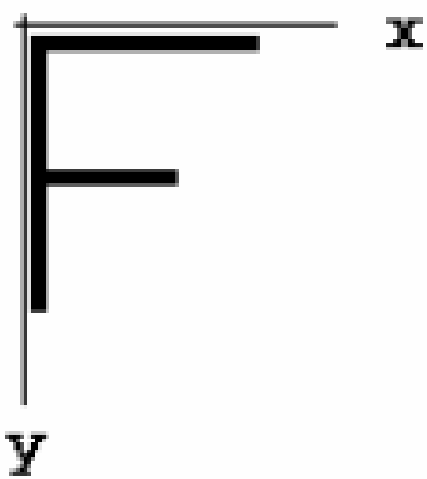
00111000
01101100
11000110
11000110
11111110
11000110
11000110
00000000

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)

Stroked Characters

- 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



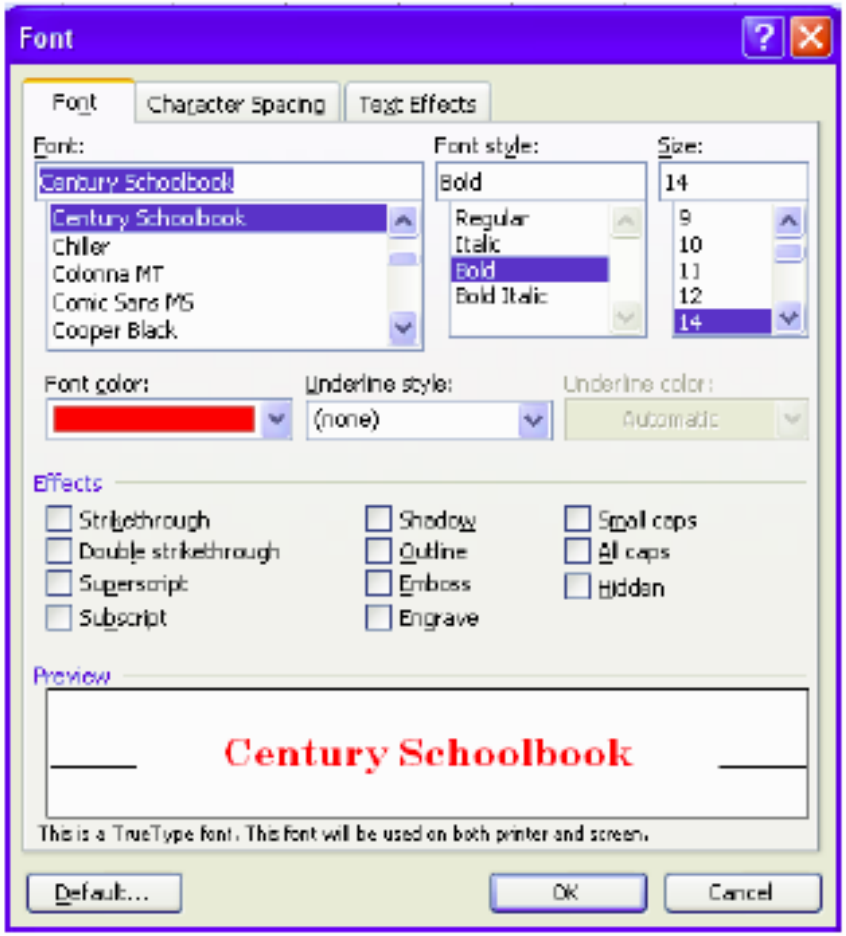

```
Strokes:

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

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

So, between these two representation:

- Bitmapped are faster to draw than outline fonts
- Outline produce higher quality characters.
- Outline take less space.



Area Filling

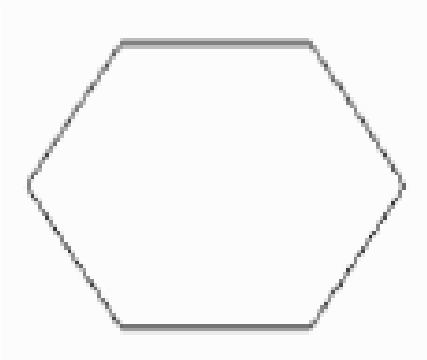
- Filling Rectangles, Polygons, Filling Ellipse, Arcs, Pattern Filling, Thick Primitives, Line Style and Pen Style.

Area-Filled Attributes

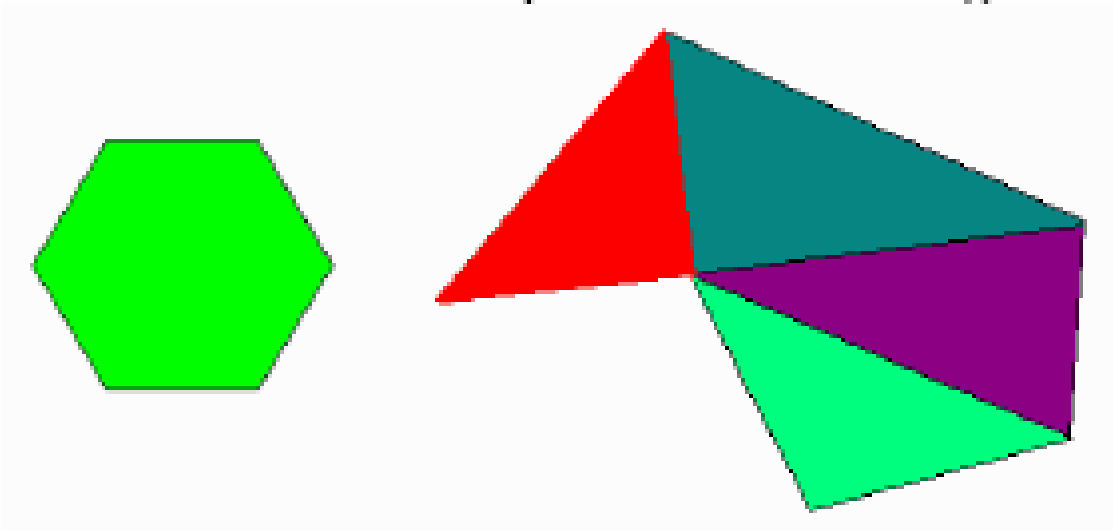
- Option for filling a defined region is whether solid, pattern and colors.

Fill Styles

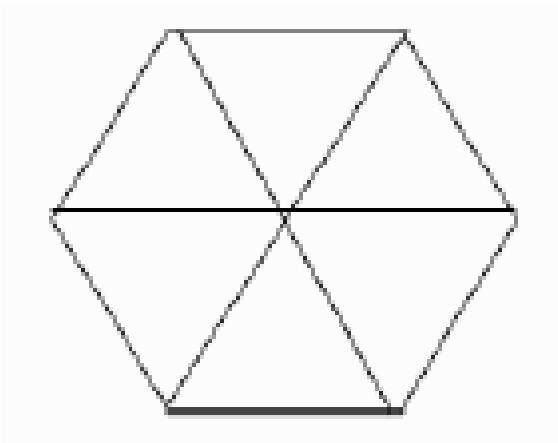
- Three basic fill styles are: hollow with color border. interior color is same with background



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



- pattern; control by other table

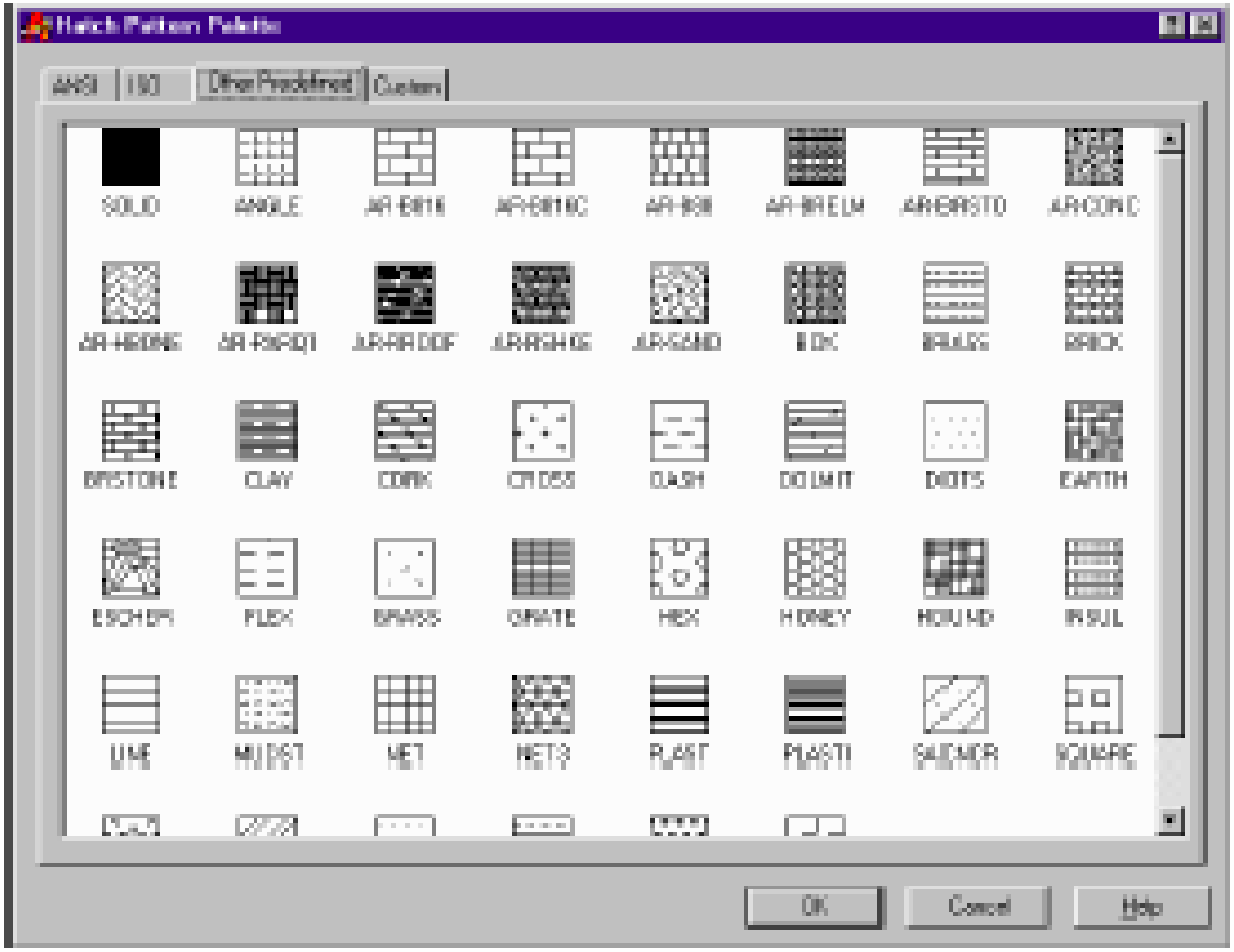


- hatch - diagonal hatch fill or diagonal cross-hatch Fill.

So to set the pattern: ex command is

- <u>SetPatternRepresentation (*ws ,pi, nx, ny, cp*)</u>
  Where:
  ws is polygon edges
  pi is set the pattern index
  cp is a 2-dimensional array color code of nx and ny.

Example


Hatch Pattern Palette
ANSI
ISO
Other Predefined
Custom
SOLID
ANGLE
BRICK
BRASS
CLAY
CORK
CROSS
DASH
DOTS
EARTH
FLEX
HEX
HONEY
INSUL
LINE
MUDST
NET
PLAST
SQUARE
OK
Cancel
Help

Area Fill (cont'd)

- Important for any closed output primitive
  - Polygons, Circles, Ellipses, etc.
- Attributes:
  - fill color
  - fill pattern
- 2 Types of area fill algorithms:
  - Boundary/Flood Fill Algorithms
  - Scanline Algorithms