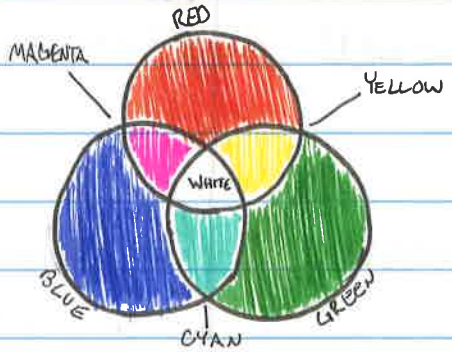
graphics

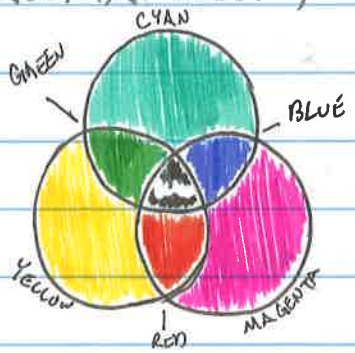
color models

RGB: RED, GREEN, BLUE



- PRIMARY COLORS OF LIGHT
- USED MOST IN COMPUTER SCREENS

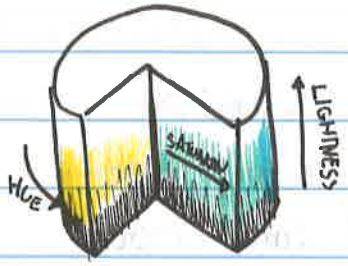
CMYK: CYAN, MAGENTA, YELLOW, BLACK



- PRIMARY COLORS OF PIGMENT
- COLORS USED IN PRINTERS

HSV: Hue, Saturation, Value

HSL: Hue, Saturation, Lightness



YC_bC_r: Luminance, Chrominance Blue, Chrominance Red

file formats

GIF: Graphics Interchange Format

- Developed by Compuserve
- Designed for 8-bit color maps
- Mostly used for animations now

JPEG: Joint Photographic Experts Group

- Semi-formal standard
- Modeled on human vision
- Uses (Y, C_b, C_r) color model
- Good compression for realistic images

PNG: Portable Network Graphics

- Designed to replace GIF
- More advanced compresson than GIF
- Supports multiple RGB models

PBM: Portable Bit Map

- Uncompressed universal format
- Intermediate format for conversions + manipulations

PGM- Portable Gray Map

PPM- Portable Pixmap

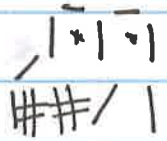
PNM- Portable Anh Map

PAM- Portable Arbitrary Map

raster image representation / display

* 2D ARRAY OF PIXELS
* PIXEL: PICTURE ELEMENT
* PIXEL VALUE
  - 0/1, Off/On, BLACK/WHITE
  - 0-n, Shades of GRAY
  - (0-n, 0-n, 0-n) RGB VALUES
* FRAME BUFFER for DISPLAY
  - B: BASE ADDRESS OF FRAME BUFFER
  - W: width of a row
  - Address of a PIXEL (c,r): B+rW+c

ascii art



* USE ASCII characters for pixels
* Eg - = + * / # for increasing darkness
* USE overstrike for more density and shading

image manipulations

* SELECT COMPONENT IMAGES
* DETERMINE IMAGE ALIGNMENT
* ADJUST COMPONENT IMAGE LEVELS
* ENHANCE SHARPNESS

finding alignment

* REPEAT FOR ALL POSSIBLE ALIGNMENTS
  - COMPUTE CORRELATION OF 2 IMAGES IN AN AREA OF OVERLAP
  - RECORD RELATIVE POSITION IF HIGHEST CORRELATION FOUND
  - MOVE TO NEXT POSSIBLE ALIGNMENT
* SOMETIMES IMPROVED BY EDGE DETECTION
* DONE IN STAGES OF INCREASING RESOLUTION

adjusting image levels

S_1: Sum of pixel values on the first image along edge of overlap

S_2: Sum of pixel values on the second image along edge of overlap

Multiply all pixel values of the second image by $\frac{S_1}{S_2}$

unsharp masking

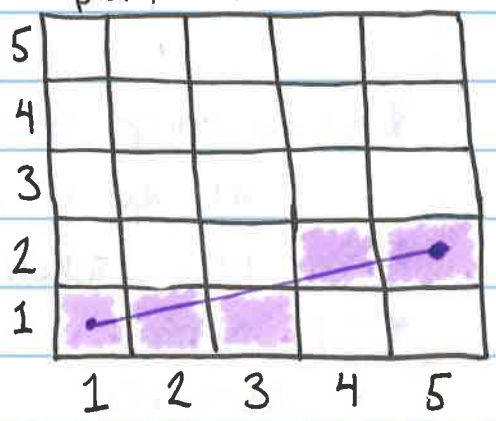
* SOME IMAGE UNSHARPNESS: SUM of sharp image and FILTERED BLURRING
* CORRECTION:
  - COMPUTE BLURRED IMAGE B BY MULTIPLYING A BLURRING MATRIX A
  - FOR EACH PIXEL IN THE IMAGE I COMPUTE THE NEW IMAGE VALUE AS $I_{ij} - \alpha B_{ij}$ FOR SOME VALUE OF α

bresenhams algorithm

```
plotLine(x0, y0, x1, y1)
    dx = x1 - x0;
    dy = y1 - y0;

    y = y0;
    e = 0;
    for (x = x0, x < x1, x++)
        plot(x, y);
        e = e + dy;
        if (2 · e > dx)
            y = y + 1;
            e = e - dx;
```

plotLine(1, 1, 5, 2);



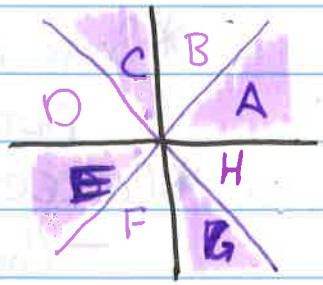
dx = 4 dy = 1

e = 0, 1, 2, 3, 4

y = 1, 2,

x = 1, 2, 3, 4, 5

* BRESENHAM CAN PLOT ANY LINE LESS THAN 45° EASILY



CASES

LETTER	ANGLE	ALTERATION
A	0°-45°	(x,y)
B	45°-90°	(y,x)
C	90°-135°	(-y,x)
D	135°-180°	(-x,y)
E	180°-225°	(-x,-y)
F	225°-270°	(-y,-x)
G	270°-315°	(y,-x)
H	315°-0°	(x,-y)

* WHEN THESE CASES ARRISE YOU CAN CHECK FOR THE IN AN IF STATEMENT
* PUT THE ALTERED COORDINATES IN THE FUNCTION
* THE OUTPUT COORDINATES WOULD THEN BE ALTERED SO SWITCH THEM BACK

Ray tracing

- LIKE MODELING A PINHOLE CAMERA
- FOR EACH PIXEL ON THE "FILM"
  - PROJECT A RAY THROUGH THE PINHOLE
  - FIND THE INTERSECTION OF THE RAY WITH THE OBJECT IN THE IMAGE
  - SET THE PIXEL TO THE COLOR AND ILLUMINATION OF THE INTERSECTION POINT ON THE OBJECT