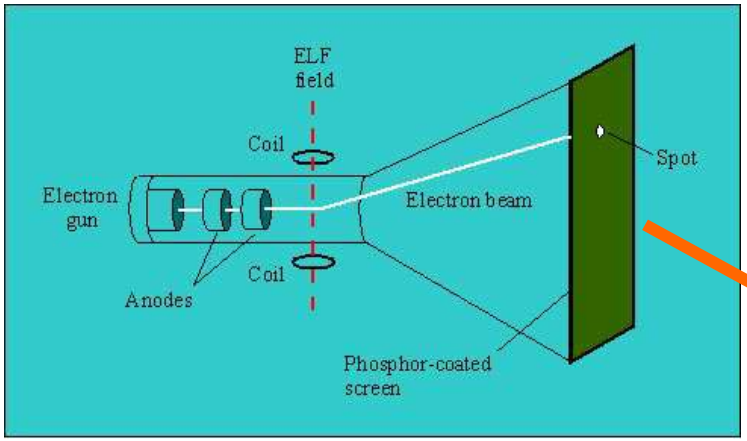
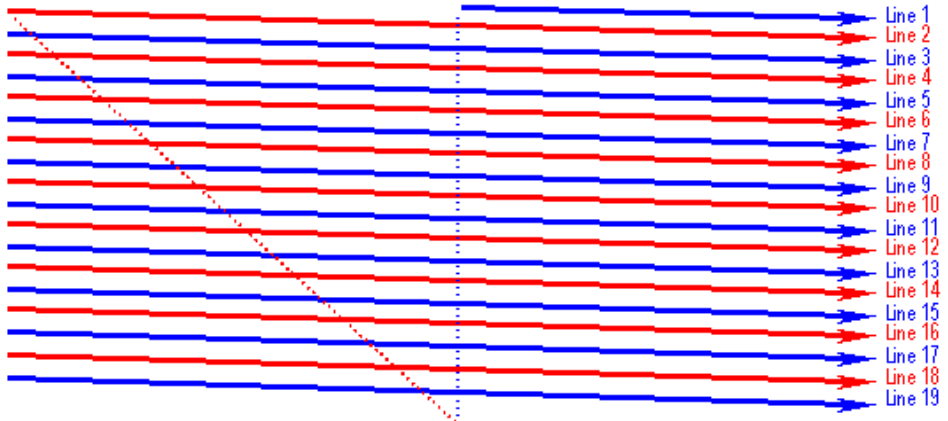
CSC418 Computer Graphics

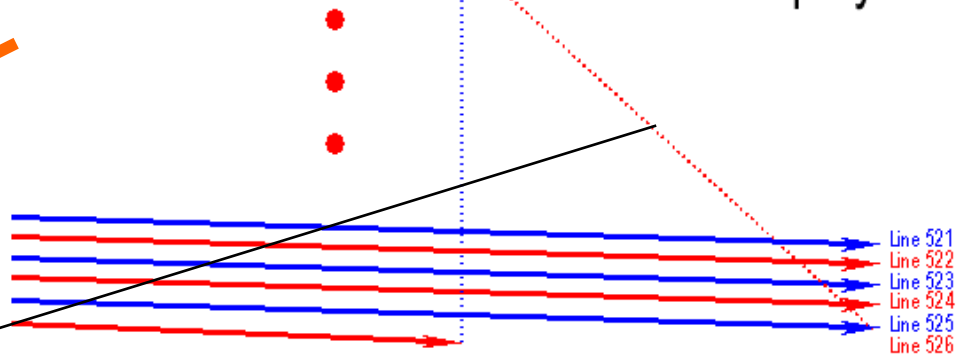
- Display Technology
- 2D modeling primitive equations
- Drawing lines

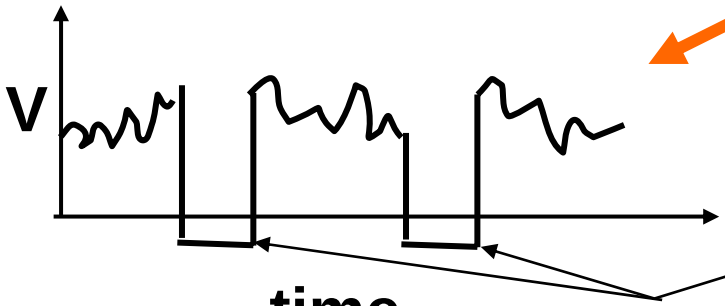
Raster Displays I





Raster Scan Pattern of Interlaced Display

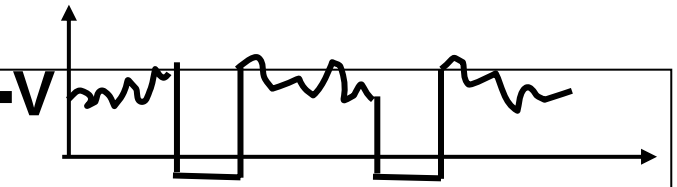




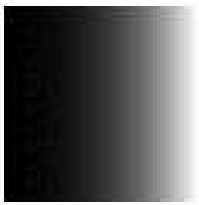
Electron beam Intensity

Raster Displays II

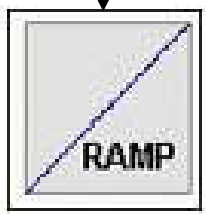
Gamma correction



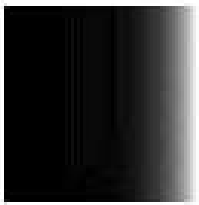
Electron beam Intensity



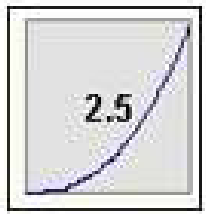
Sample Input to Monitor



Graph of Input



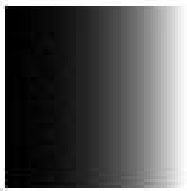
Output from Monitor



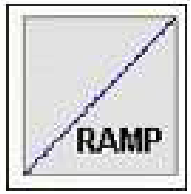
Graph of Output **L = V ^ 2.5**

Raster Displays II

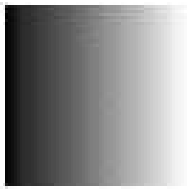
Gamma correction



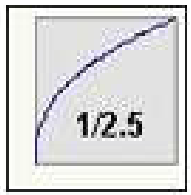
Sample Input



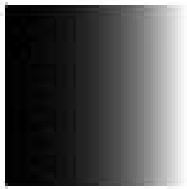
Graph of Input



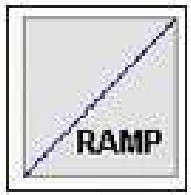
Gamma Corrected Input



Graph of Correction $L' = L^{(1/2.5)}$



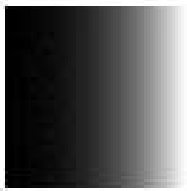
Monitor Output



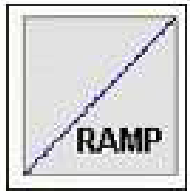
Graph of Output

Raster Displays II

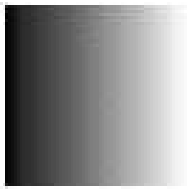
Gamma correction



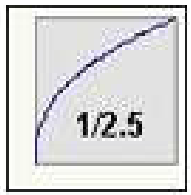
Sample Input



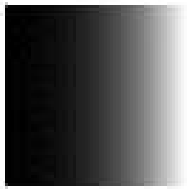
Graph of Input



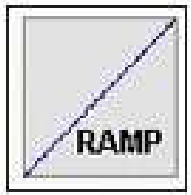
Gamma Corrected Input



Graph of Correction $L' = L^{(1/2.5)}$



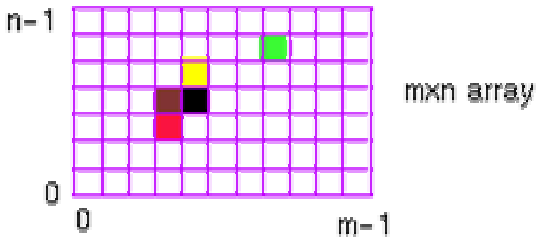
Monitor Output

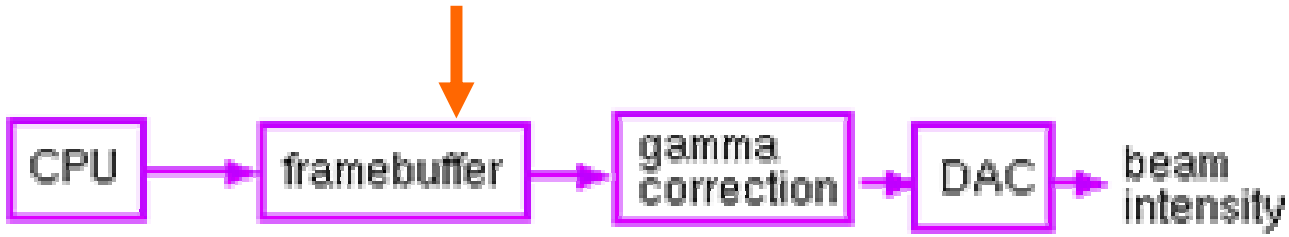


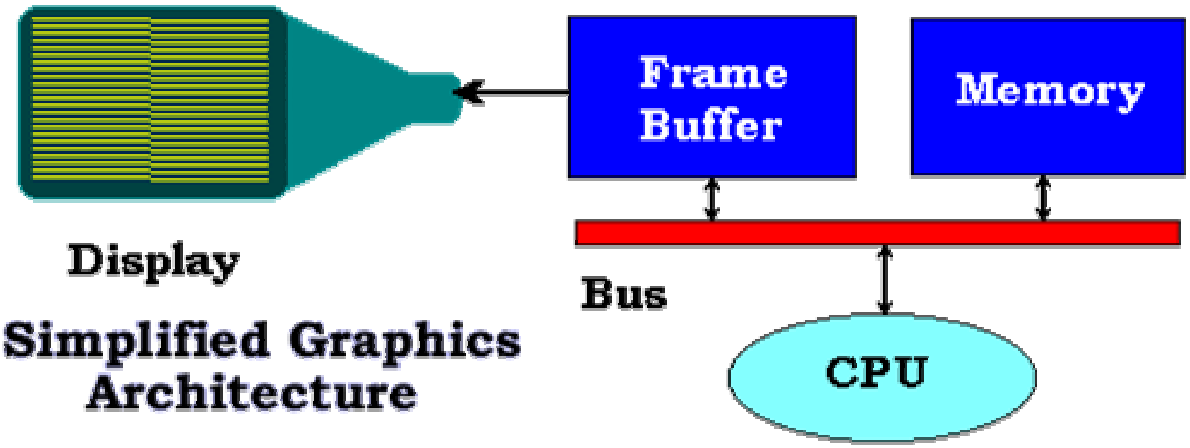
Graph of Output

Display Architecture

Frame Buffer

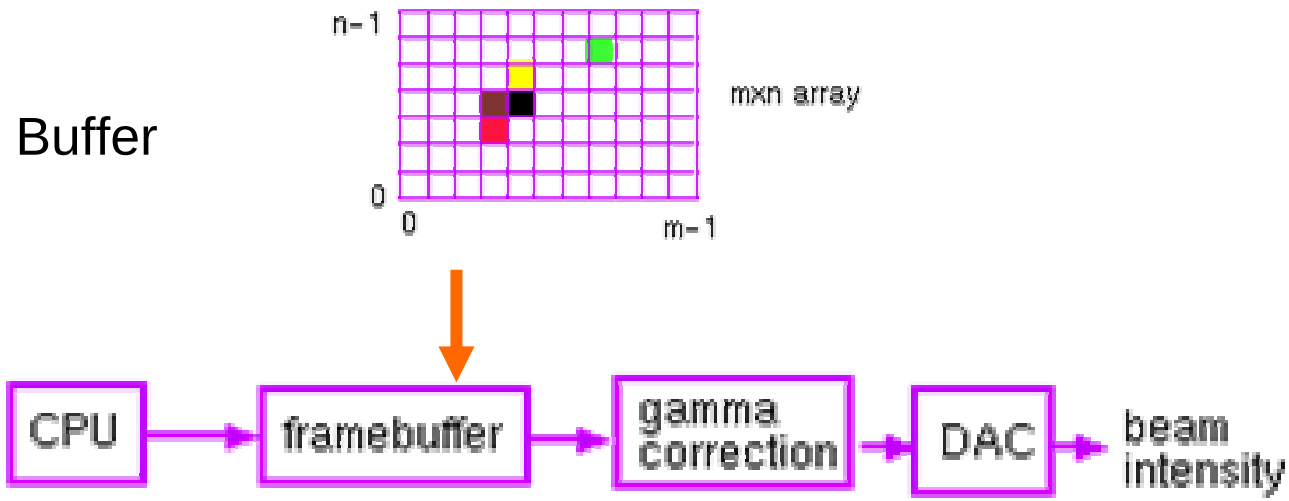


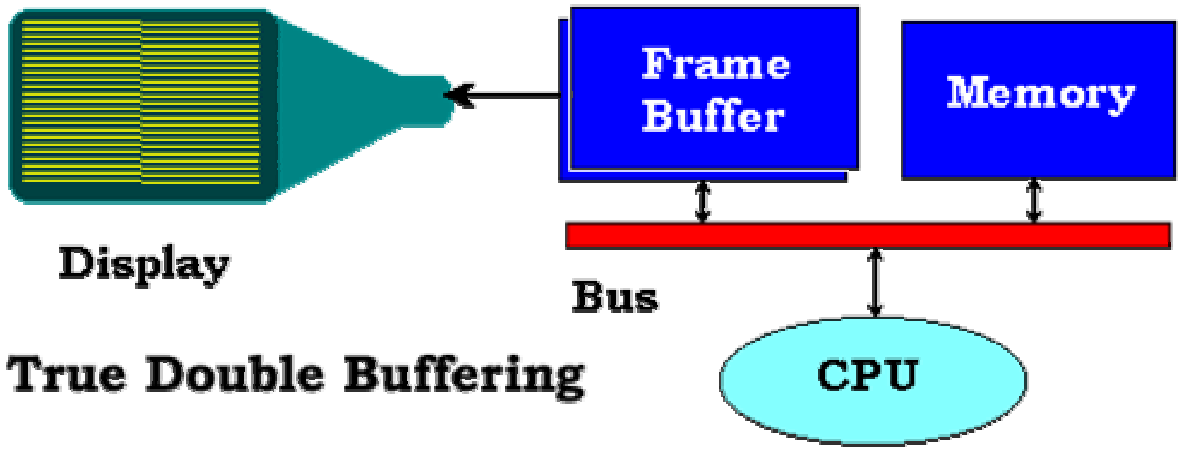




Display Architecture

Double Buffer

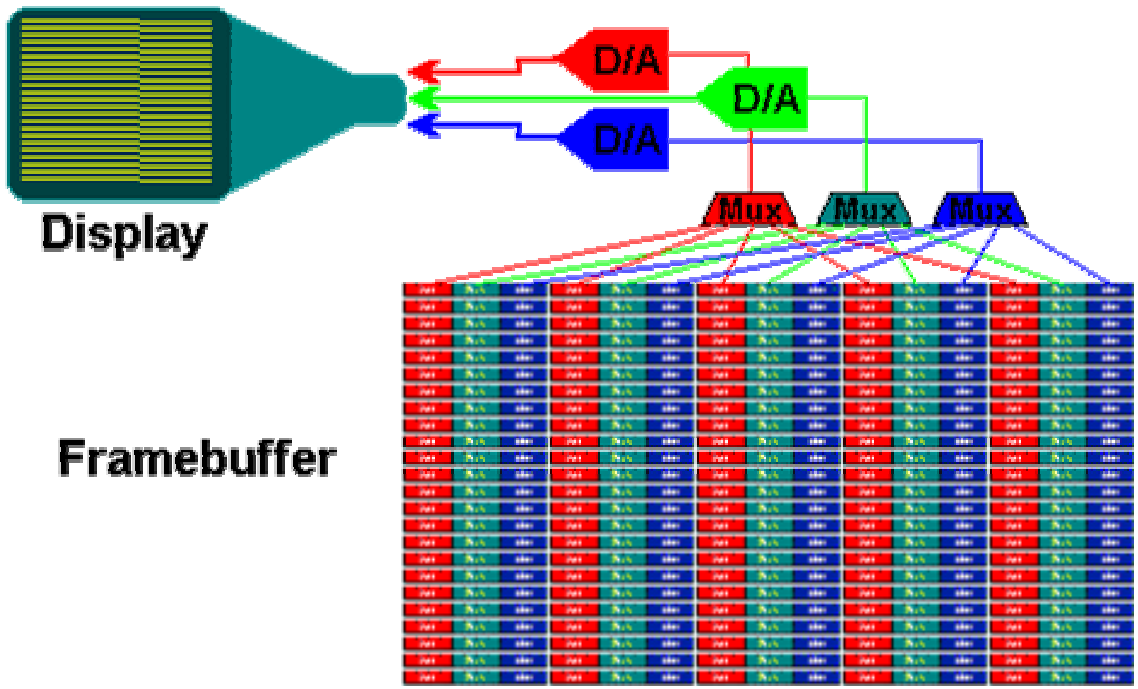




True Double Buffering

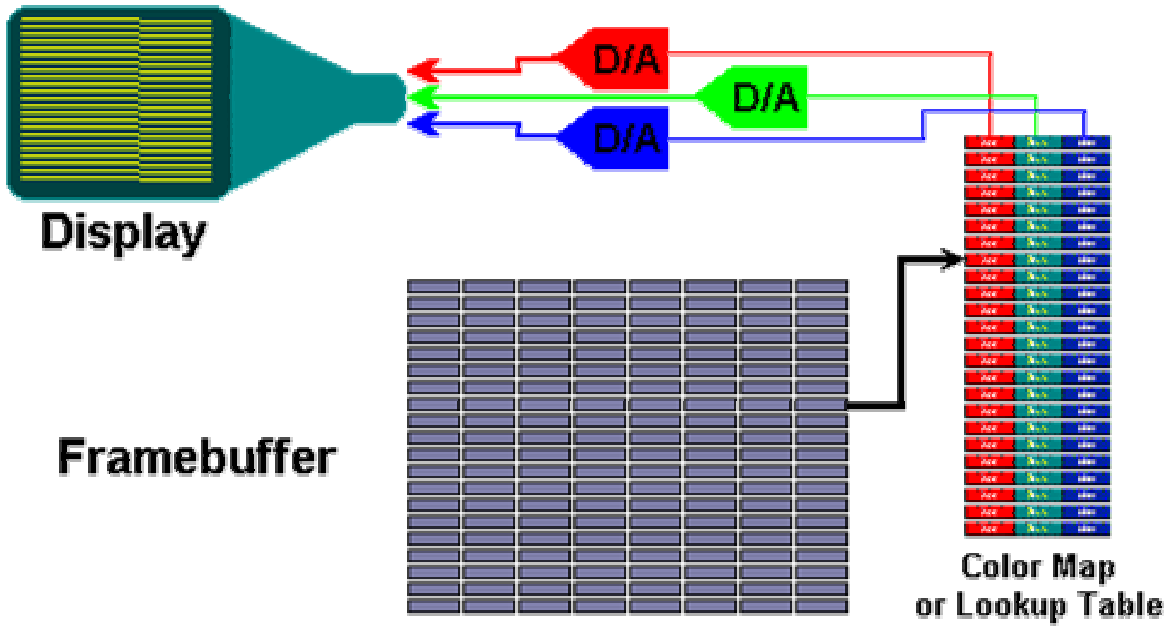
Display Architecture II

True Color Frame Buffer : 8 bits per pixel RGB



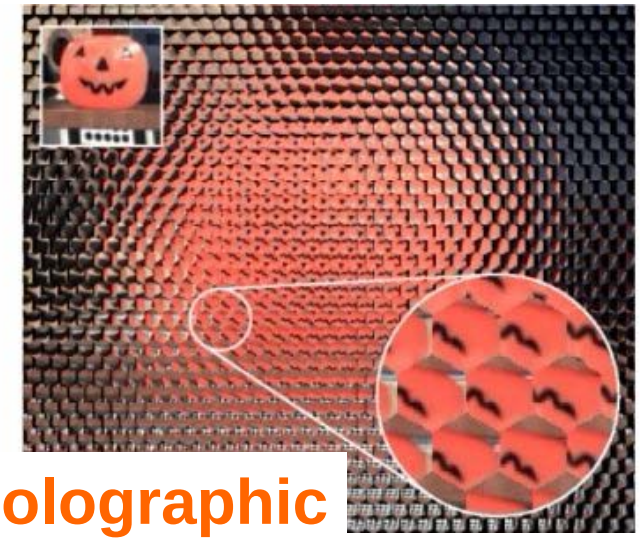
Display Architecture II

Indexed Color Frame Buffer : 8 bit index to color map



Display Devices II

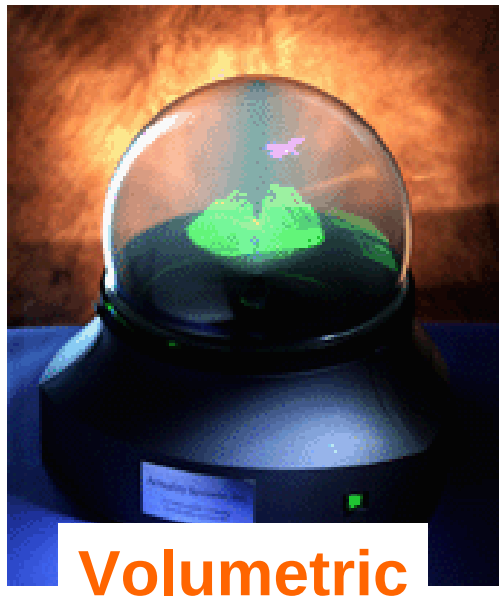
Plasma



Holographic

Immersive

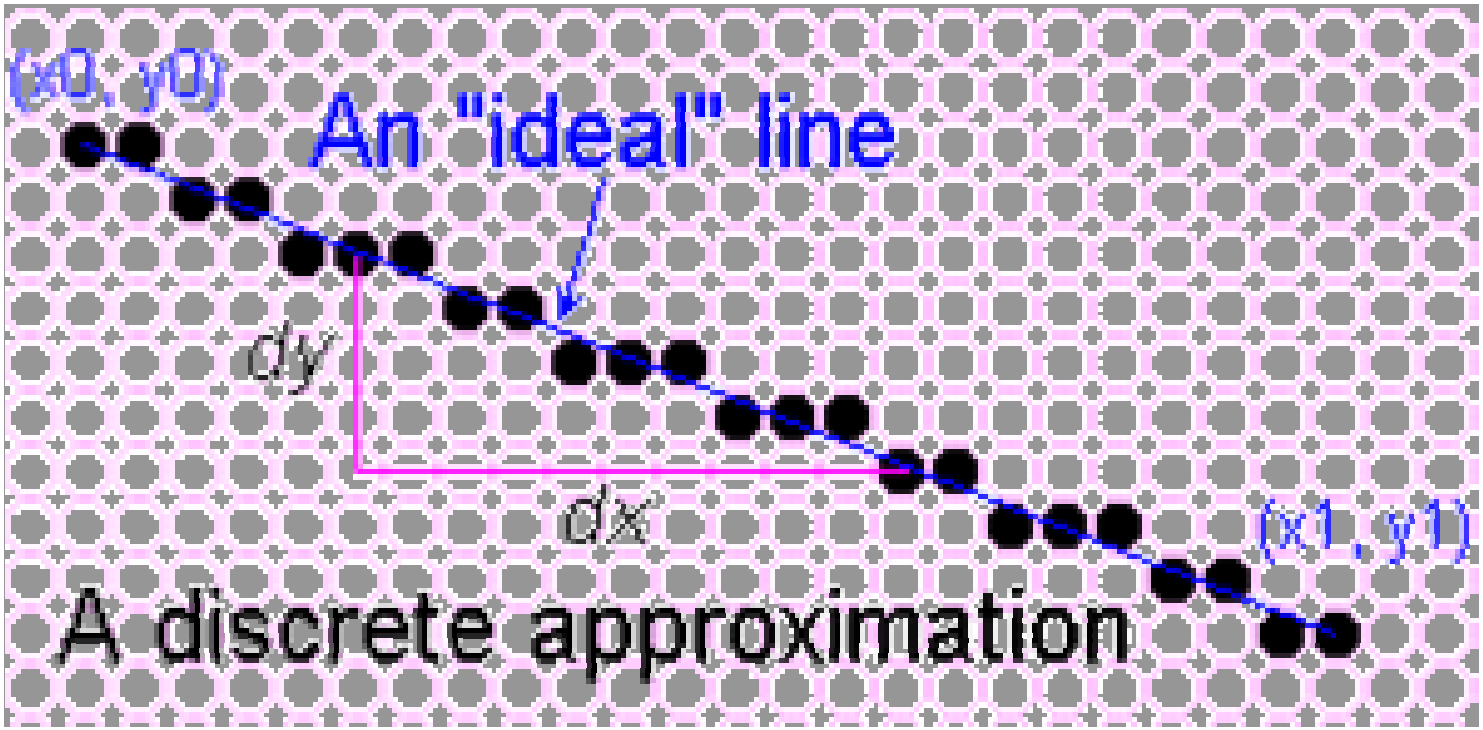
Head-mounted



Volumetric

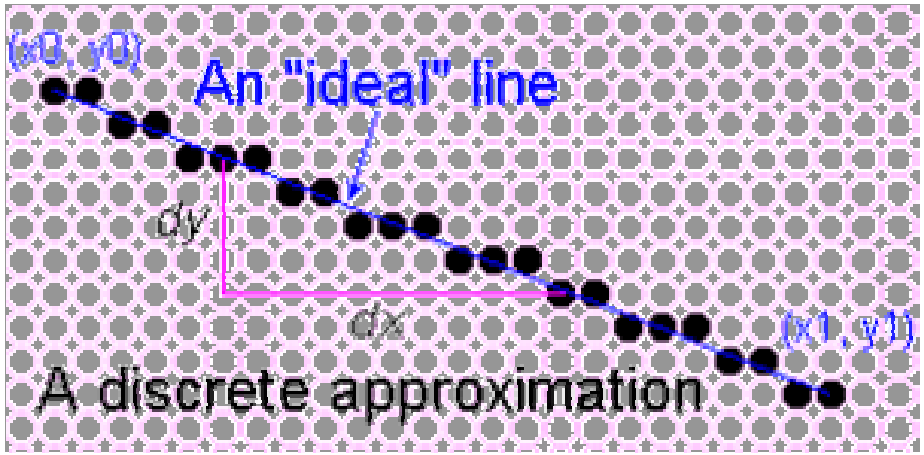
Line Drawing

What is the best line line we can draw?



Line Drawing

What is the best line line we can draw?



The best we can do is a discrete approximation of an ideal line.

Important line qualities:

- **Continuous appearence**
- **Uniform thickness and brightness**
- **Accuracy (Turn on the pixels nearest the ideal line)**
- **Speed (How fast is the line generated)**

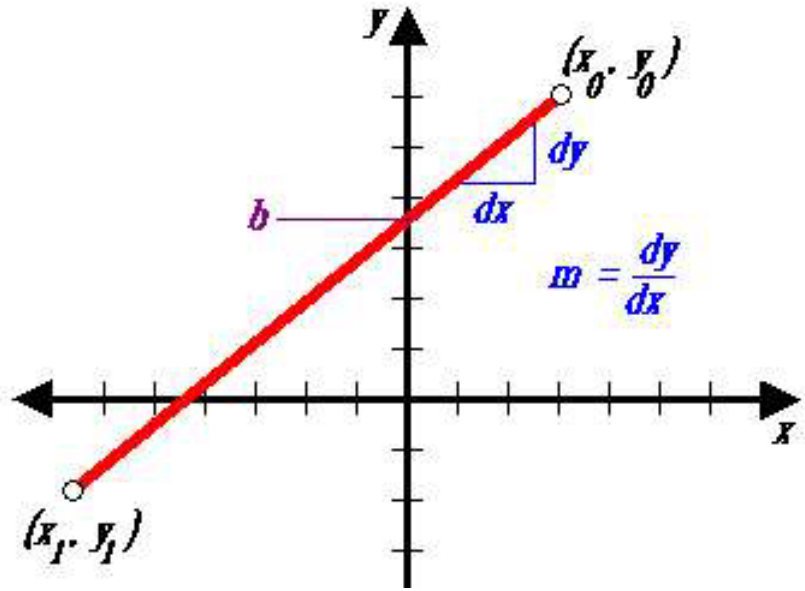
Equation of a Line

Explicit : $y = mx + b$

Parametric :

$x(t) = x_0 + (x_1 - x_0)*t$

$y(t) = y_0 + (y_1 - y_0)*t$



$P = P_0 + (P_1 - P_0)*t$

$P = P_0*(1-t) + P_1*t$ **(weighted sum)**

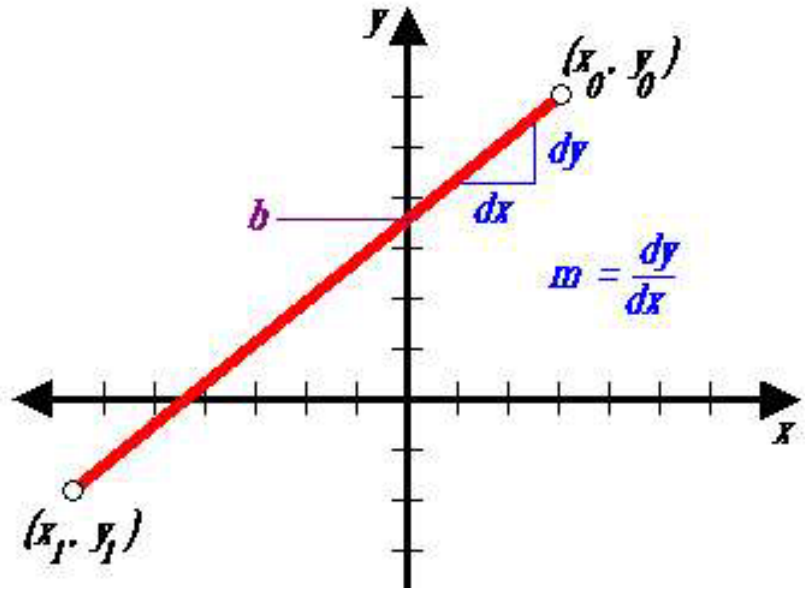
Implicit : $(x-x_0)dy - (y-y_0)dx = 0$

Algorithm I

Explicit form:

$$y = dy/dx * (x - x_0) + y_0$$

```
float y;
int x;
for ( x=x0; x<=x1; x++)
{
    y= y0 + (x-x0)*(y1-y0)/(x1-x0);
    setpixel (x, round(y));
 }
```

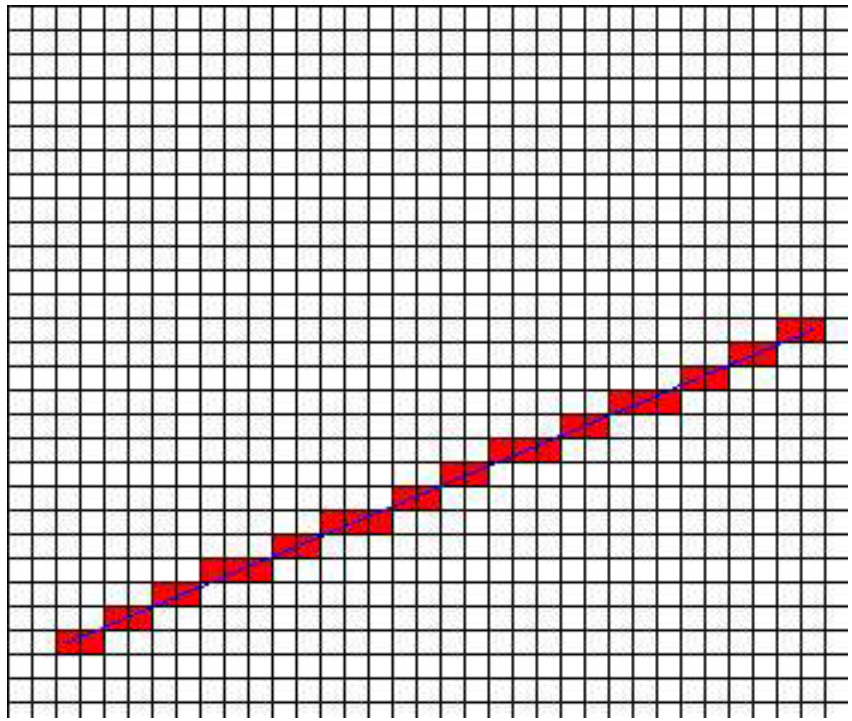


Algorithm I

Explicit form:

$$y = dy/dx * (x - x_0) + y_0$$

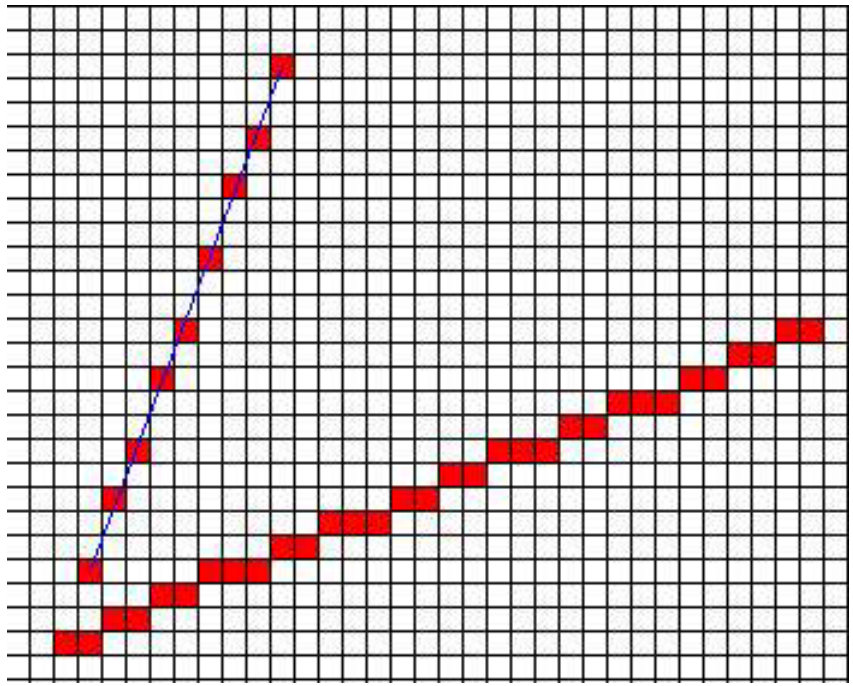
```
float y;
int x;
dx = x1-x0; dy = y1 – y0;
m = dy/dx;
y= y1 + 0.5;
for ( x=x0; x<=x1; x++)
{
    setpixel (x, floor(y));
    y= y + m;
 }
```



Algorithm I

DDA (Digital Differential Analyzer)

```
float y;
int x;
dx = x1-x0; dy = y1 – y0;
m = dy/dx;
y= y1 + 0.5;
for ( x=x0; x<=x1; x++)
{
    setpixel (x, floor(y));
    y= y + m;
 }
```



Algorithm II

Bresenham Algorithm

- **Assume |line slope <1**
- **Slope is rational (ratio of two integers).** $m = (y1 - y0) / (x1 - x0)$
- **The incremental part of the algorthim never generates a new y that is more than one unit away from the old one**

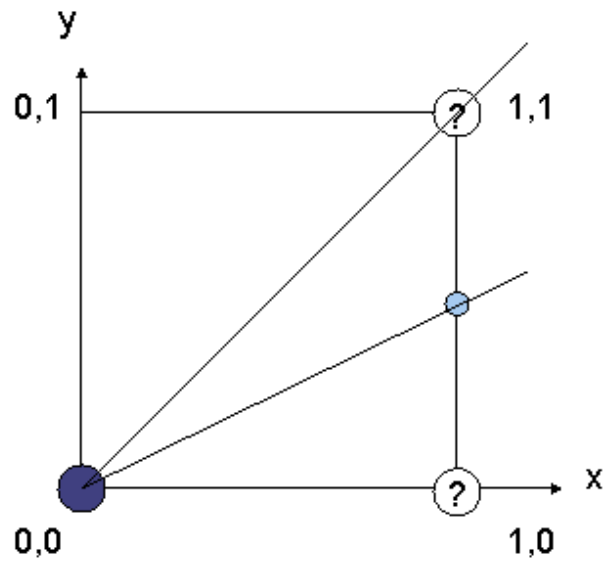
  (because the slope is always less than one) $y_{i+1} = y_i + m$

Algorithm II

Bresenham Algorithm Geometric Interpretation

Distance of midpt from line

= dy- ½*dx



Algorithm II

Bresenham Algorithm

Implicit View

$F(x,y) = (x-x0)dy - (y-y0)dx$

$F(x+1,y + 0.5) = F(x,y) + dy - 0.5\,dx$

$2\,F(x+1,y+ 0.5) = d = 2F(x,y) + 2dy - dx$

$F(x+1,y) = F(x,y) + dy \qquad d' = d + 2dy$

$F(x+1,y+1) = F(x,y) + dy - dx \qquad d' = d + 2dy - 2dx$

CSC418 Computer Graphics

Next Lecture….

- **Polygons**
  - **Triangulation**
  - **Scan conversion**
  - **Convex/Concave**
  - **clipping)**
- **2D affine transformations and properties, Homogeneous coordinates**