SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

Introduction to Computer Graphics and Visualization

By

Prof. Vaibhav P. Vasani

Assistant Professor

Department of Computer Engineering

K. J. Somaiya College of Engineering

Somaiya Vidyavihar University

Somaiya TRUST

Graphics Pipeline

Modeling

- Polygons
- Constructive Solid Geometry
  - union, intersection, difference of solids
- Parametric Surfaces
  - **f**(**u**) = x(u,v), y(u,v), z(u,v)
- Implicit Surfaces
  - locus of solutions of f(**x**) = f(x,y,z) = 0
- Subdivision Surfaces
  - sketch at board
- Particle Systems
- Volumes

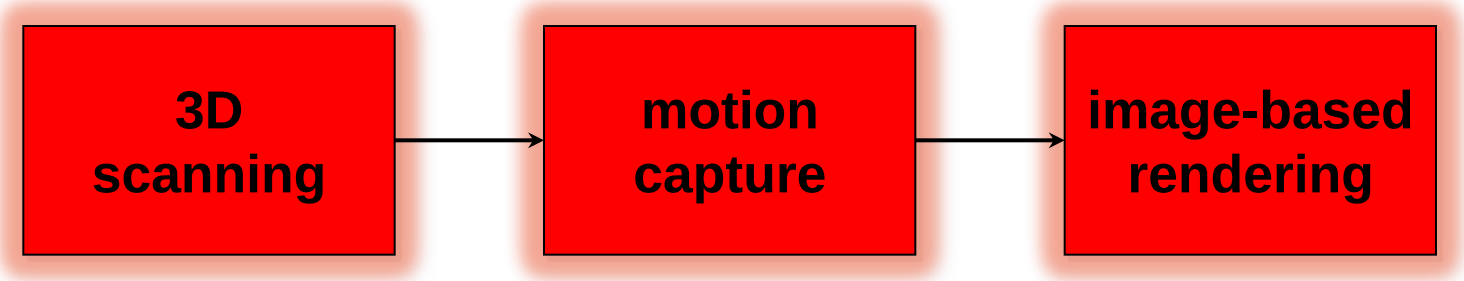
Animation

- scripted
- key-frame interpolation
- inverse kinematics
- dynamics

The graphics pipeline

the traditional pipeline

the new pipeline?



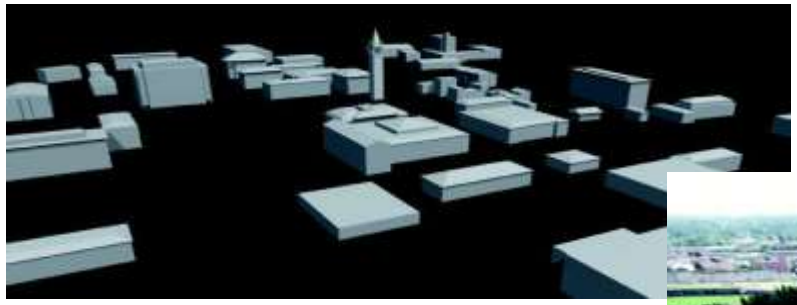
Rendering

- Rendering or image synthesis is the process of generating a photorealistic image from a 2D or 3D model using computer program.

Example: Rendering

Image-Based Rendering

1. Appearance



Exactly What Can We Capture From images?

2. Geometry

3. Reflectance & Illumination

4. Motion

IBR Pros and Cons

- Pros
  - Modest computation compared to classical C.G.
  - Cost independent of scene complexity
  - Imagery from real or virtual scenes
- Cons
  - Static scene geometry
  - Fixed lighting
  - Fixed look-from or look-at point

Appearance Modeling






Appearance Models

- **Appearance models** in computer graphics and vision try to answer these questions
  - Why does the sky **appear** blue?
  - Why does wet sand **appear** darker than dry sand?
  - Why do iridescent surfaces (CD-ROM, butterflies, hummingbird) wings) **appear** to have different colors when viewed in different directions ?
  - Why do old and weathered surfaces **appear** different from new ones?
  - Why do rusted surfaces **appear** different from un-rusted ones?

Real Time Rendering

- Photo-realistic rendering does NOT care how long it takes
- If we have a technique that renders realistic sea shells but it takes days, still use it
- Some applications require images to be displayed relatively quickly = Real-time rendering
- Examples:
  - Games
  - Flight simulators
  - Virtual reality
  - Augmented reality, etc

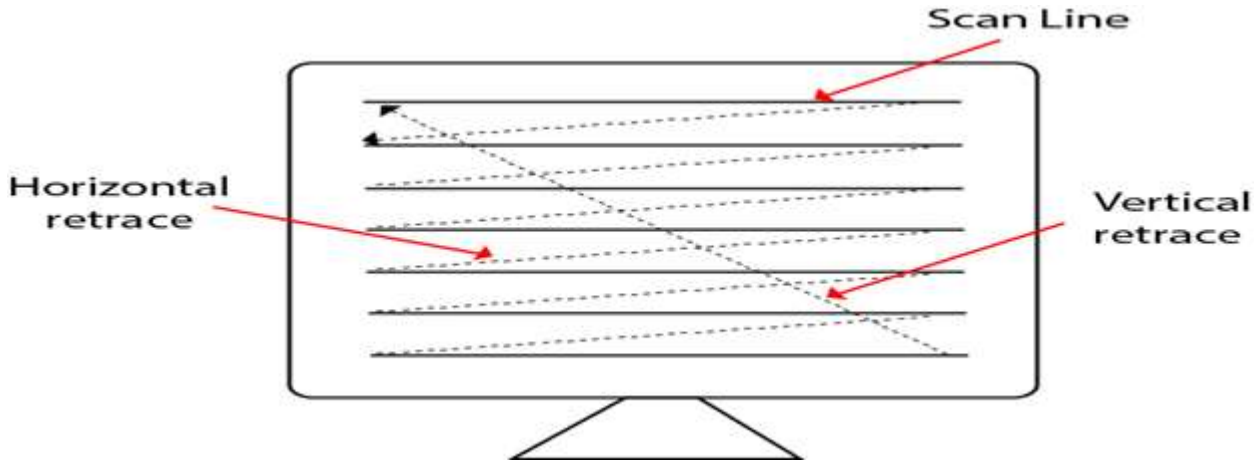
What is Real-Time Rendering?

- Purist may argue that real-time rendering should happen instantaneously (too strict)
- We can relax this a little
- Rendering speed measured in Frames Per Second (fps)
- Frank Cho, electronics arts, used algorithms must run in at most 30 fps (minimum)
- About 72 fps guarantees that user:
  - becomes immersed in graphics experience
  - interacts freely
  - No distraction of waiting for rendering to complete
- So, we can say 15 fps upwards is real-time
- All images produced must have feel of 3D graphics

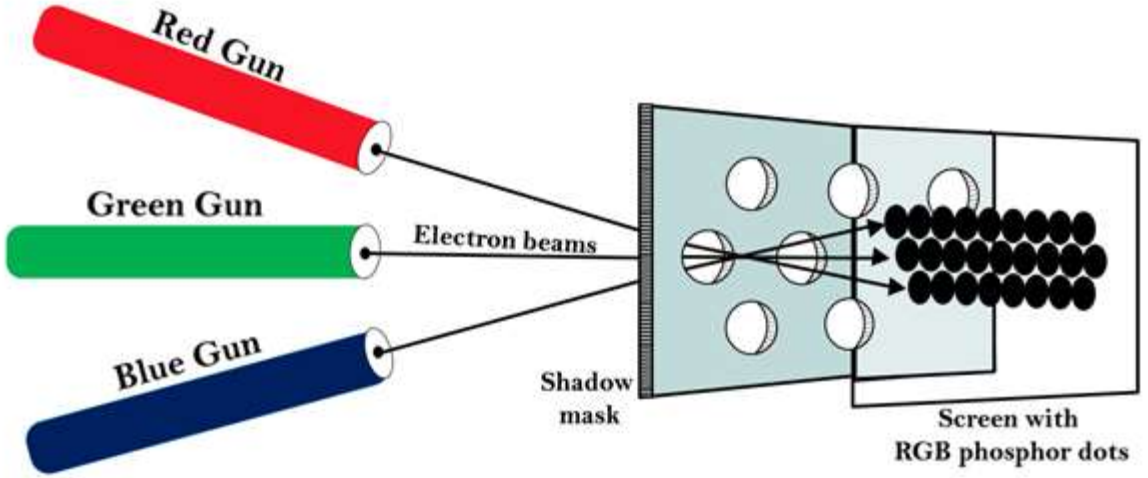
Real-Time Rendering

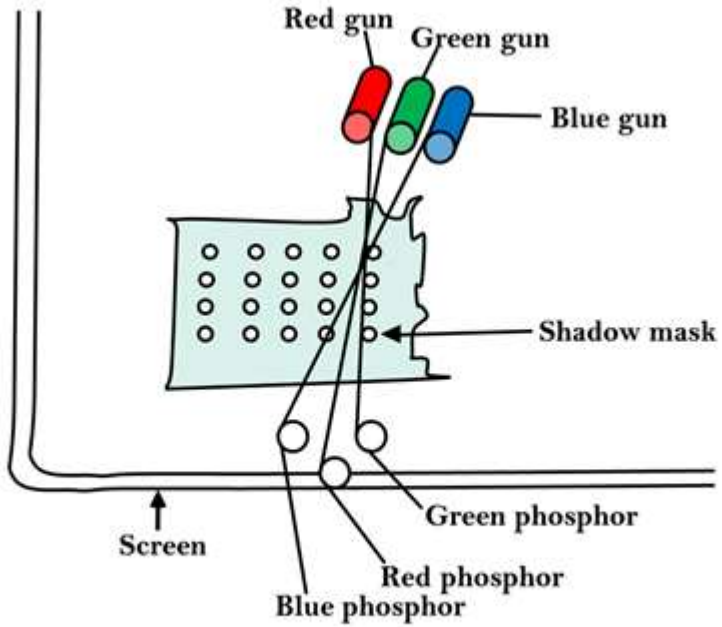
- How can we achieve real-time speeds
  - Pre-process graphics models (simplify, replace polygons with textures, etc)
  - Graphics Hardware acceleration
- Graphics Hardware:
  - Previously, SGI was king
  - Today: 3D graphics cards from ATI, Nvidia on PCs
  - 3Dfx Voodoo 1 was first card in 1996
    - beginning of real-time graphics era?
  - Most of advances in real time graphics are due to innovation in graphics cards
    - Chip on card also called Graphics Processing Unit (GPU)
    - Speed: GPUs renders graphics faster CPUs
    - Programmability: Flexibility

Raster Scan



- **Types of Scanning or travelling of beam in Raster Scan**
  - Interlaced Scanning
    - Interlaced scanning, each horizontal line of the screen is traced from top to bottom. Due to which fading of display of object may occur.
  - Non-Interlaced Scanning
    - In this first of all odd numbered lines are traced or visited by an electron beam, then in the next circle, even number of lines are located.





The Shadow mask CRT

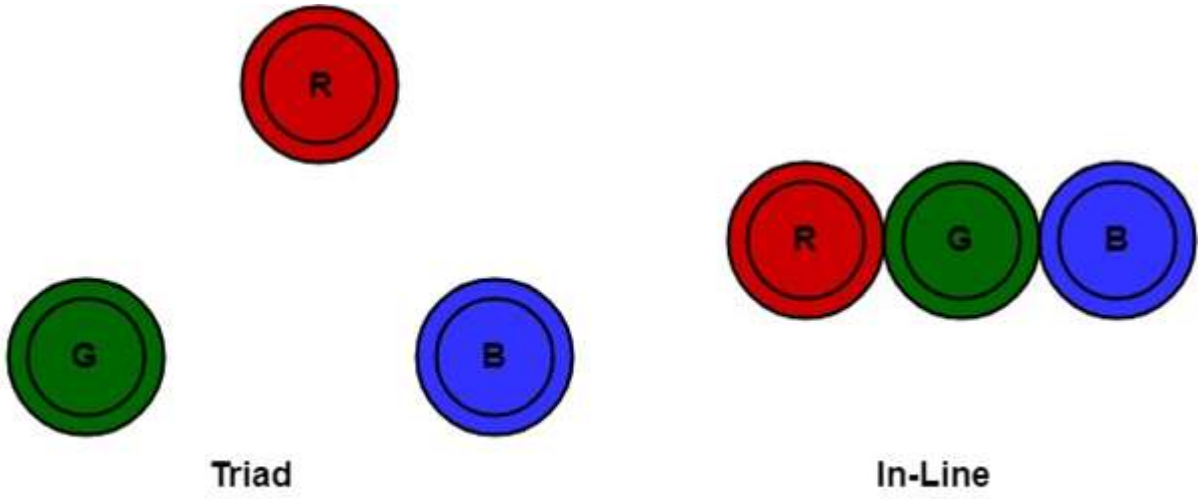
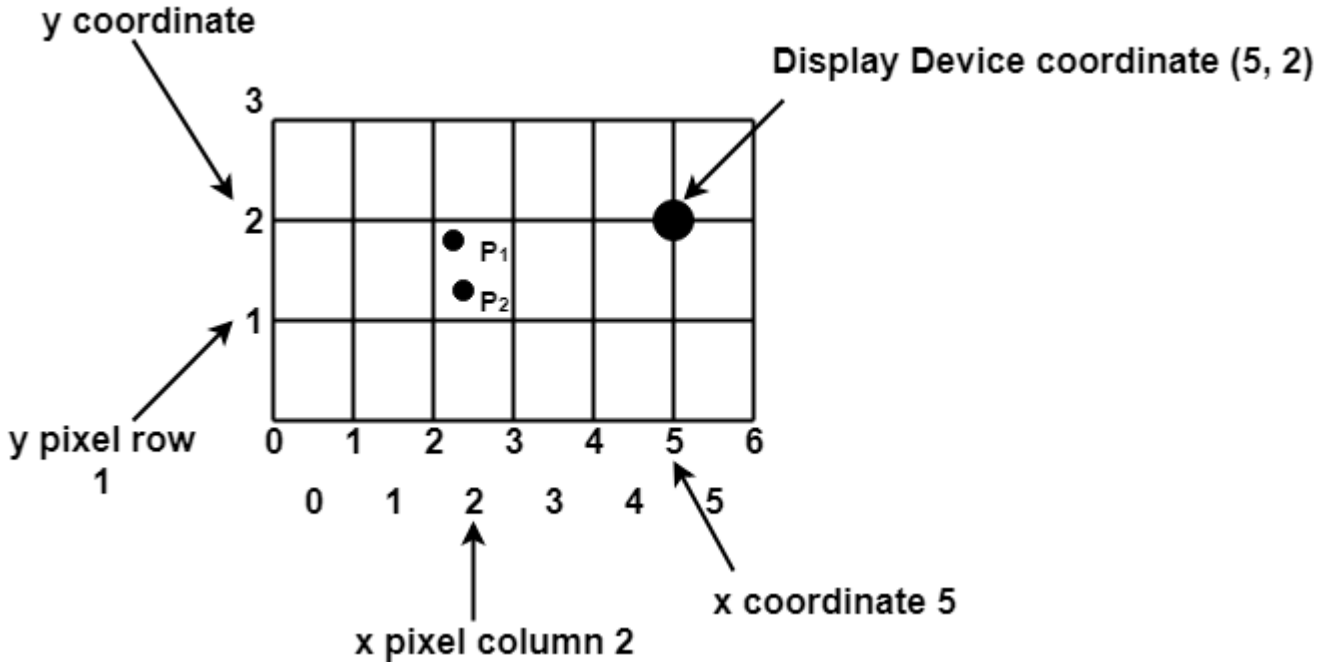


Fig: Triad-and -in-line arrangements of red, green and blue electron guns of CRT for color monitors.

Scan Converting a Point

- Each pixel on the graphics display does not represent a mathematical point.
- A region which theoretically can contain an infinite number of points.
- Scan-Converting a point involves illuminating the pixel that contains the point.

Example: Display coordinates points as shown in fig would both be represented by pixel (2, 1). In general, a point p (x, y) is represented by the integer part of x & the integer part of y that is pixels [(INT (x), INT (y).



DDA

- Digital Differential Analyzer A.K.A. Incremental method
- equation of the straight line :- **$y = mx + c$**
- Here, **m** is the slope of **(x_1, y_1)** and **(x_2, y_2).**
- **$m = (y_2 - y_1)/ (x_2 - x_1)$**

- Now, we consider one point (x_k, y_k) and (x_{k+1}, y_{k+1}) as the next point.
- Then the slope $m = (y_{k+1} - y_k) / (x_{k+1} - x_k)$
- Now, we have to find the slope between the starting point and ending point. There can be following three cases to discuss:

Case 1: If **m < 1**

Then **x** coordinate tends to the Unit interval.

$$x_{k+1} = x_k + 1$$
$$y_{k+1} = y_k + m$$

Case 2: If **m > 1**

Then **y** coordinate tends to the Unit interval.

$$y_{k+1} = y_k + 1$$
$$x_{k+1} = x_k + 1/m$$

Case 3: If **m = 1**

Then **x and y** coordinate tend to the Unit interval.

$$x_{k+1} = x_k + 1$$
$$y_{k+1} = y_k + 1$$

Step 1: Start.

Step 2: We consider Starting point as **(x_1, y_1)**, and ending point **(x_2, y_2).**

Step 3: Now, we have to calculate **dx** and **dy.**

$$dx = x_2 - x_1$$

$$dy = y_2 - y_1$$

$$m = dy/dx$$

Step 4: Now, we calculate three cases.

If **m < 1**

Then **x** change in Unit Interval

y moves with deviation

$$(x_{k+1}, y_{k+1}) = (x_k+1, y_k+m)$$

If **m > 1**

Then **x** moves with deviation

y change in Unit Interval

$$(x_{k+1}, y_{k+1}) = (x_k+1/m, y_k+1)$$

If **m = 1**

Then **x** moves in Unit Interval

y moves in Unit Interval

$$(x_{k+1}, y_{k+1}) = (x_k+1, y_k+1)$$

Step 5: We will repeat step 4 until we find the ending point of the line.

Step 6: Stop.

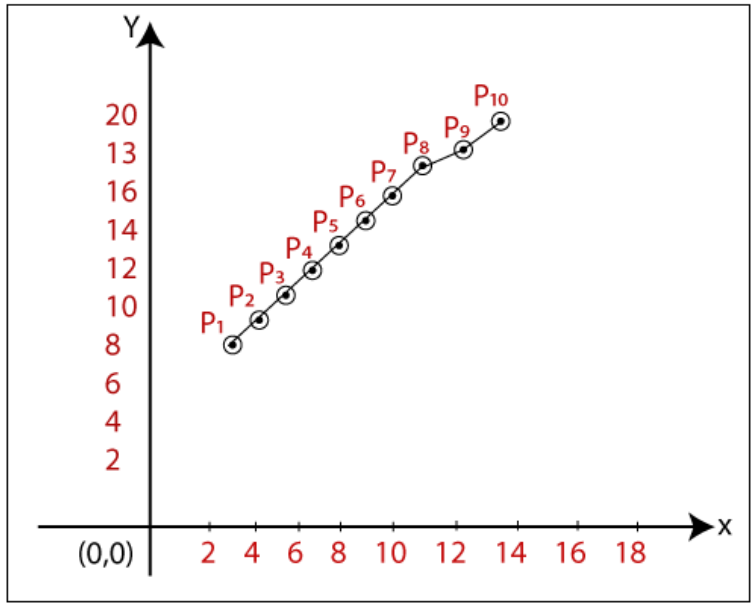
Example: A line has a starting point (1,7) and ending point (11,17). Apply the Digital Differential Analyzer algorithm to plot a line.

- **Solution:** We have two coordinates,
- Starting Point = (x_1, y_1) = **(1,7)**
- Ending Point = (x_2, y_2) = **(11,17)**
- **Step 1:** First, we calculate **dx**, **dy** and **m.**
- $dx = x_2 - x_1 = 11-1 = 10$
- $dy = y_2 - y_1 = 17-7 = 10$
- $m = dy/dx = 10/10 = 1$
- **Step 2:** Now, we calculate the number of steps.
- $dx = dy = 10$
- Then, the number of steps = **10**
- **Step 3:** Weget **m = 1,** Third case is satisfied.
- Now move to next step.

Example: A line has a starting point (1,7) and ending point (11,17). Apply the Digital Differential Analyzer algorithm to plot a line.

x_k	y_k	x_{k+1}	y_{k+1}	(x_{k+1}, y_{k+1})
1	7	2	8	(2, 8)
		3	9	(3, 9)
		4	10	(4, 10)
		5	11	(5, 11)
		6	12	(6, 12)
		7	13	(7, 13)
		8	14	(8, 14)
		9	15	(9, 15)
		10	16	(10, 16)
		11	17	(11, 17)

Step 4: We will repeat step 3 until we get the endpoints of the line.
Step 5: Stop.



Advantages of Digital Differential Analyzer

- It is a simple algorithm to implement.
- It is a faster algorithm than the direct line equation.
- We cannot use the multiplication method in Digital Differential Analyzer.
- Digital Differential Analyzer algorithm tells us about the overflow of the point when the point changes its location.

Disadvantages of Digital Differential Analyzer

- The floating-point arithmetic implementation of the Digital Differential Analyzer is time-consuming.
- The method of round-off is also time-consuming.
- Sometimes the **point position is not accurate.**

Thank you