Computer Graphics

Lecture - 03

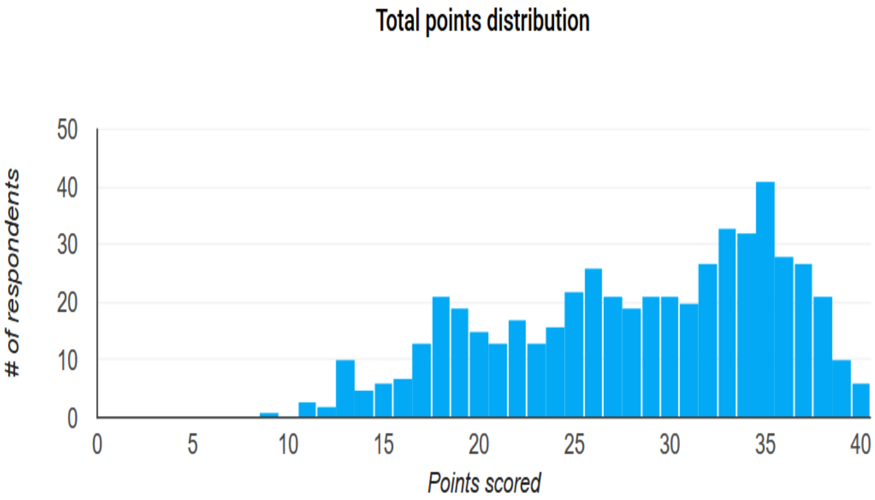
Updated Version

Haitham A. El-Ghareeb

Faculty of Computers and Information Sciences
Mansoura University
Egypt
helghareeb@mans.edu.eg

March 1, 2019

Average	Median	Range
28.27 / 40 points	29 / 40 points	9 - 40 points



1. Environmental Evolution
   - Character Displays
   - Vector
   - 2D bitmap
   - 3D Graphics
   - High-end PCs
2. Graphics Display Hardware
   - Vector
   - Raster
   - Conceptual Framework for Interactive Graphics
   - Graphics Library
3. Application Distinctions - Two Basic Paradigms
   - Sample-based Graphics
   - Geometry-based Graphics

4. Sample-based Graphics
    - Sampling an Image
    - Advantages of Sampling Images
    - Sampling Images Disadvantages

5. Geometry-Based Graphics
    - Geometric Modeling

6. Decomposition of Geometric Model
    - Hierarchical (Tree) Diagram of Nail
        - Composition of Geometric Model

Character Display

- Character Displays (1960s – now) - Figure 1 at page 6
- Display: text plus alphamosaic pseudo-graphics (ASCII art)
- Object and command specification: command-line typing
- Control over appearance: coding for text formatting (.p = paragraph, .i 5 = indent 5)
- Application control: single task

Figure: Character Display

Vector

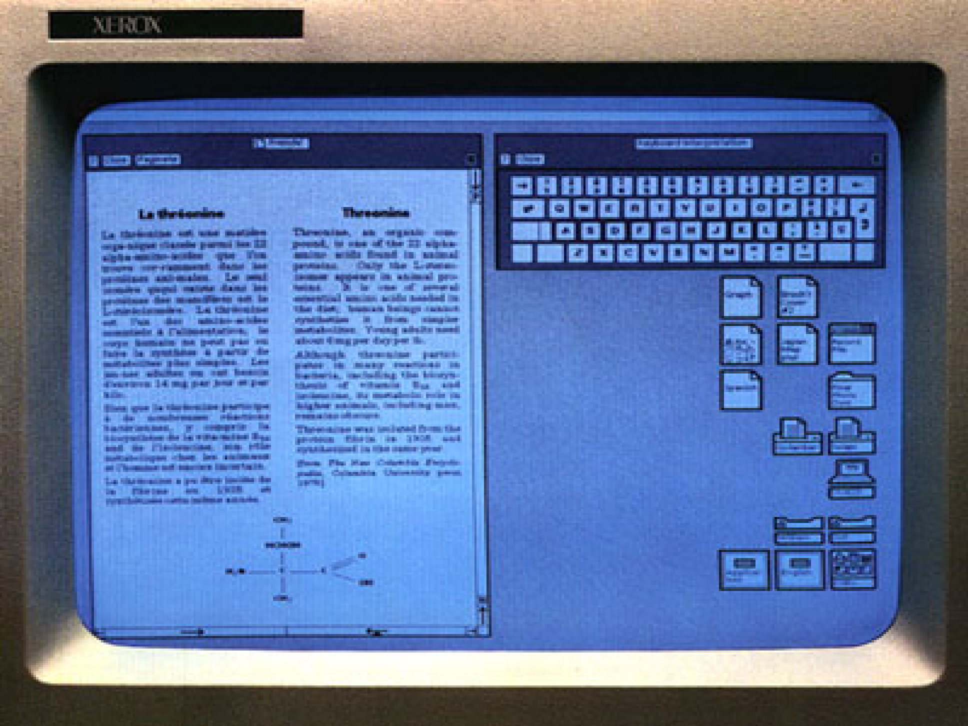
- Vector (Calligraphic, Line Drawing) - Figure **??** at page **??**
- Displays (1963 – 1980s)
- Display: line drawings and stroke text; 2D and 3D transformation hardware
- Object and command specification: command-line typing, function keys, menus
- Control over appearance: pseudo-WYSIWYG
- Application control: single or multitasked, distributed computing pioneered at Brown via mainframe host ¡-¿ minicomputer satellite
- Term "vector" graphics survives as "scalable vector graphics" library from Adobe and W3C – shapes as

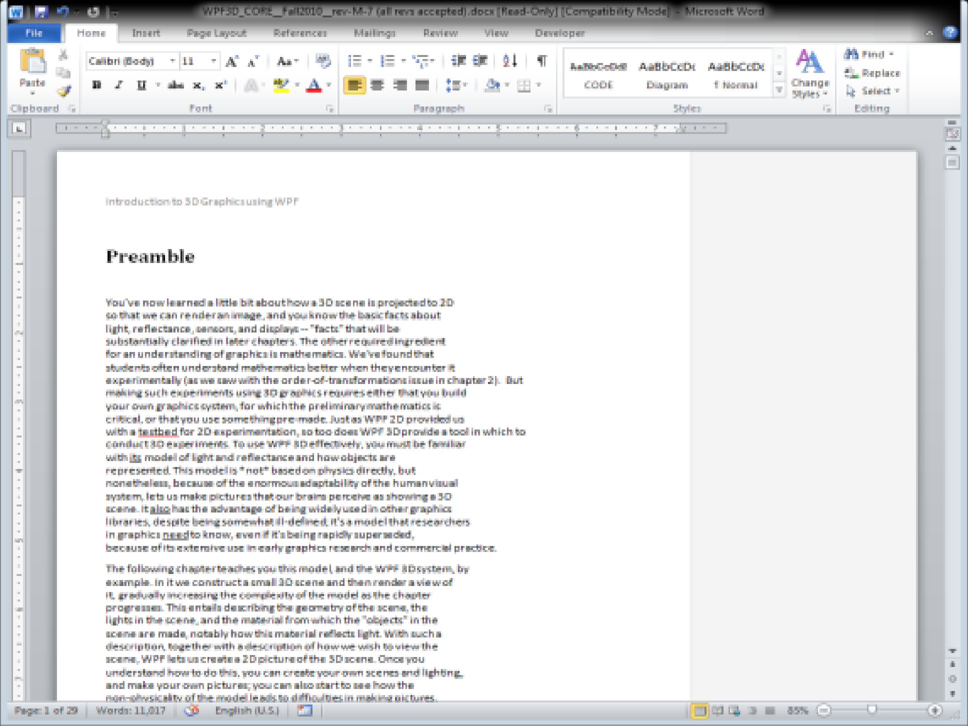
2D bitmap - I

- 2D bitmap raster displays for PCs and workstations (1972 at Xerox PARC - now)
- Display: windows, icons, legible text, "flat earth" graphics
  - Note: late 60's saw first use of raster graphics, especially for flight simulators
- Object and command specification: minimal typing via WIMP (Windows, Icons, Menus, Pointer) GUI: point-and-click selection of menu items and objects, widgets and direct manipulation (e.g., drag and drop), "messy desktop" metaphor

2D bitmap - II

- Control over appearance: WYSIWYG (which is really WYSIAYG, What You See Is All You Get)
- Application control: multi-tasking, networked client-server computation and window management (even "X terminals")
- Figure **??** presented at page **??** presents a classic WIMP interface. The technology, at its core, remains largely the same today. Figure **??** presented at **??** presents a modern WIMP interface.


XEROX
La thréonine
Threonine



Preamble

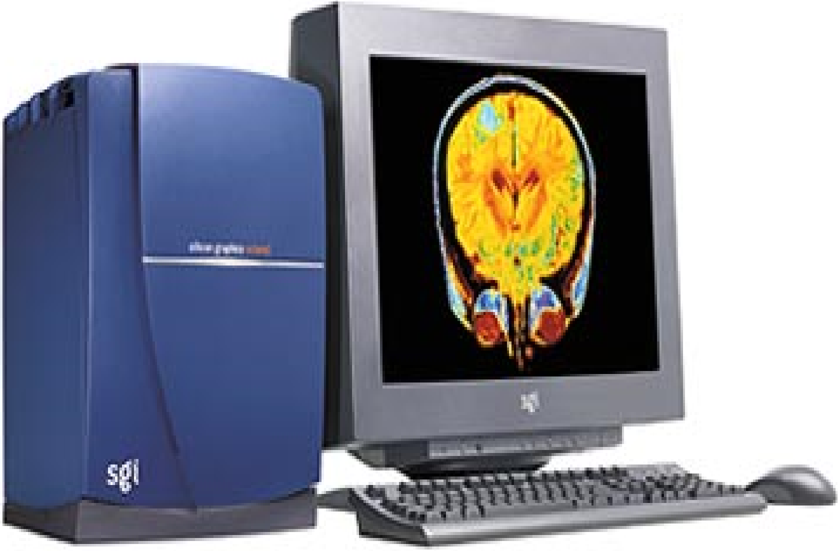
You've now learned a little bit about how a 3D scene is projected to 2D so that we can render an image, and you know the basic facts about light, reflectance, sensors, and displays -- "facts" that will be substantially clarified in later chapters. The other required ingredient for an understanding of graphics is mathematics. We've found that students often understand mathematics better when they encounter it experimentally (as we saw with the order-of-transformations issue in chapter 2). But making such experiments using 3D graphics requires either that you build your own graphics system, for which the preliminary mathematics is critical, or that you use something pre-made. Just as WPF 2D provided us with a testbed for 2D experimentation, so too does WPF 3D provide a tool in which to conduct 3D experiments. To use WPF 3D effectively, you must be familiar with its model of light and reflectance and how objects are represented. This model is *not* based on physics directly, but nonetheless, because of the enormous adaptability of the human visual system, lets us make pictures that our brains perceive as showing a 3D scene. It also has the advantage of being widely used in other graphics libraries, despite being somewhat ill-defined; it's a model that researchers in graphics need to know, even if it's being rapidly superseded, because of its extensive use in early graphics research and commercial practice.

The following chapter teaches you this model, and the WPF 3D system, by example. In it we construct a small 3D scene and then render a view of it, gradually increasing the complexity of the model as the chapter progresses. This entails describing the geometry of the scene, the lights in the scene, and the material from which the "objects" in the scene are made, notably how this material reflects light. With such a description, together with a description of how we wish to view the scene, WPF lets us create a 2D picture of the 3D scene. Once you understand how to do this, you can create your own scenes and lighting, and make your own pictures; you can also start to see how the non-physicality of the model leads to difficulties in making pictures.

3D Graphics

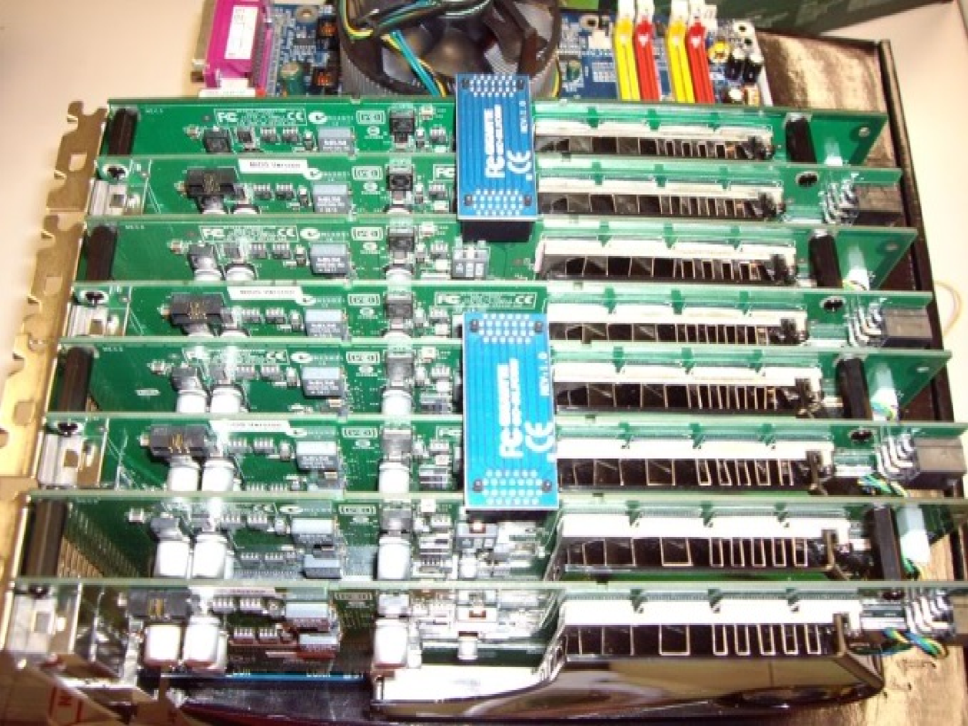
- 3D graphics workstations (1984 at SGI – now)
- Display: real-time, pseudo-realistic images of 3D scenes
- Object and command specification: 2D, 3D and N-D input devices (controlling 3+ degrees of freedom) and force feedback haptic devices for point-and-click, widgets, and direct manipulation
- Control over appearance: WYSIWYG (still WYSIAYG)
- Application control: multi-tasking, networked (client/server) computation and window management
- Graphics workstations such have been replaced with commodity hardware (GPUs)

Silicon Graphics® Octane2™



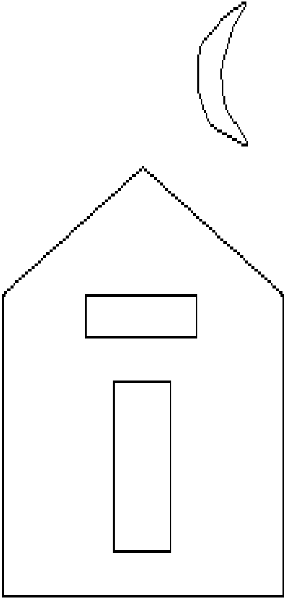
High-end PCs

- High-end PCs with hot graphics cards (nVidia GeForce™, ATI Radeon™) have supplanted graphics workstations
- Such PCs are clustered together over high speed buses or LANs to provide "scalable graphics" to drive tiled PowerWalls, Caves, etc.
- Now accessible to consumers via new technologies like NVIDIA's SLI bridge
- You can put multiple GPUs together in your computer using SLI

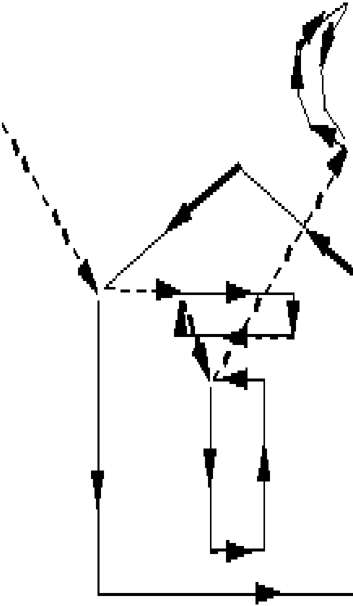


Vector

- calligraphic, stroke, random-scan
- Driven by display commands (move (x, y), char("A") , line(x, y)...)
- Survives as "scalable vector graphics"
- Figure **??** presented at page **??** presents mapping between ideal drawing and vector drawing



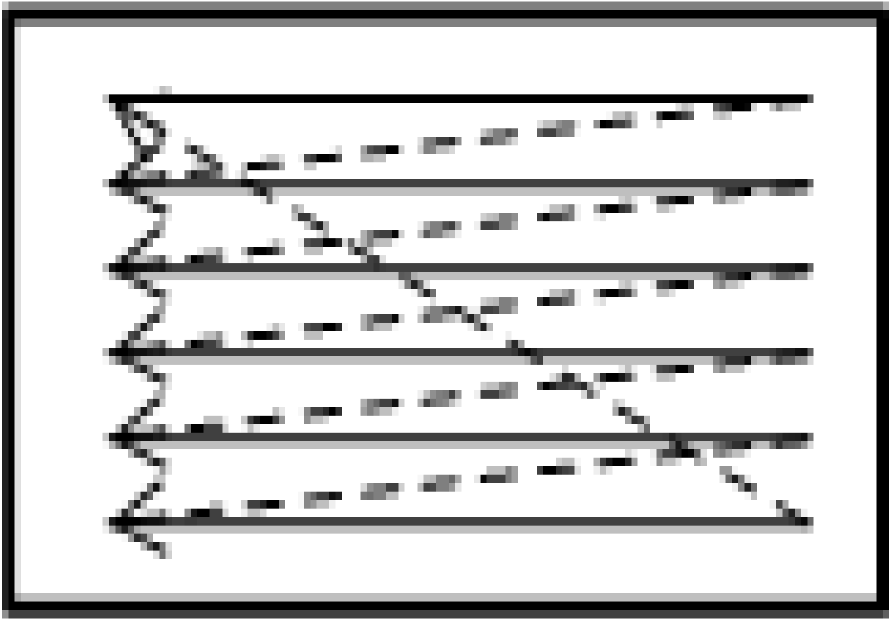
Ideal Drawing

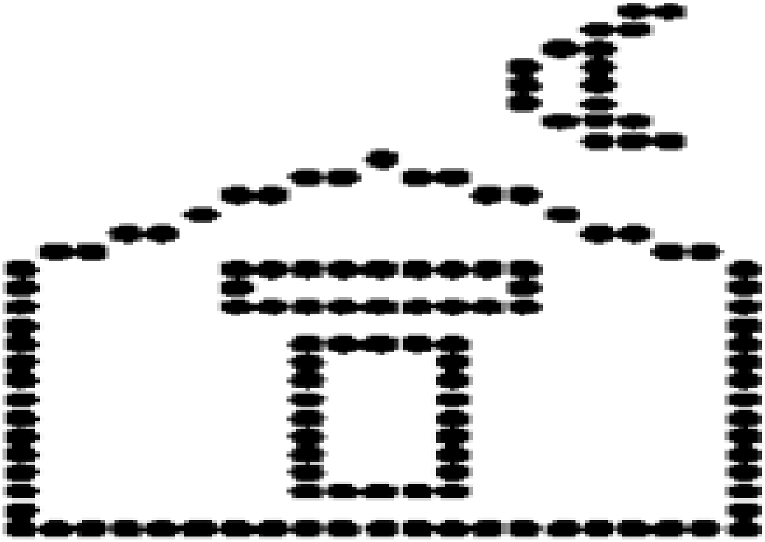


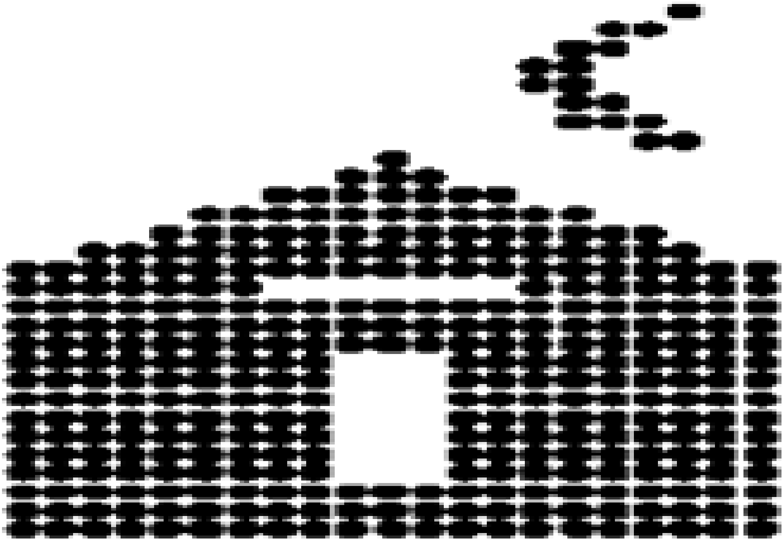
Vector Drawing

Raster

- (TV, bitmap, pixmap) used in displays and laser printers
- Driven by array of pixels (no semantics, lowest form of representation)
- Note “jaggies” (aliasing errors) due to sampling continuous primitives



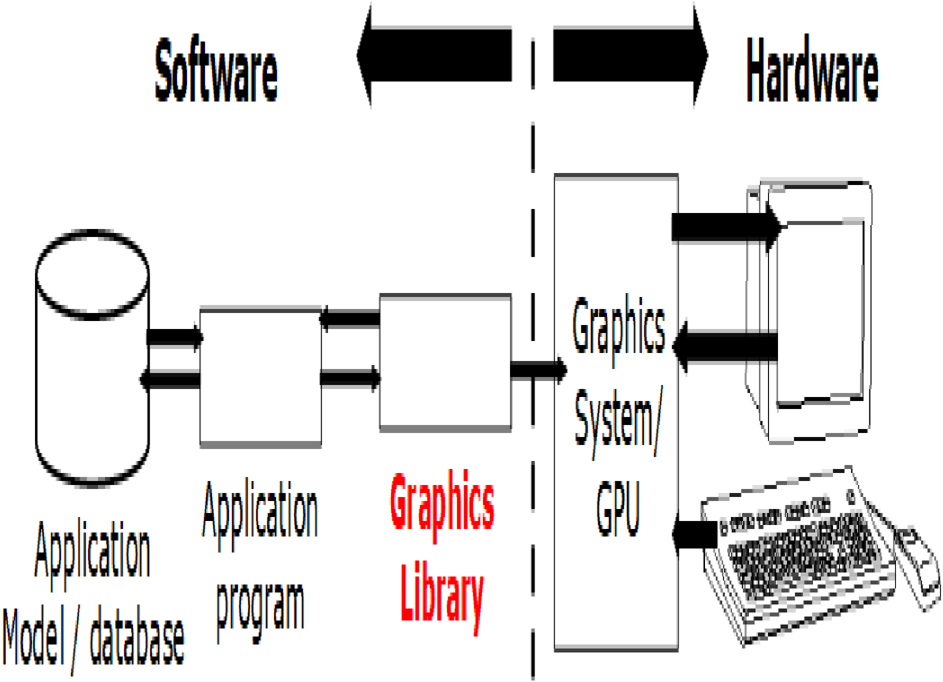

outline primitives



filled primitives

Conceptual Framework for Interactive Graphics

- Graphics library/package is intermediary between application and display hardware (Graphics System)
- Application program maps application objects to views (images) of those objects by calling on graphics library. Application model may contain lots of non-graphical data (e.g., non-geometric object properties)
- User interaction results in modification of model and/or image
- Figure **??** presented at page **??** represents such a model
- This hardware and software framework is more than 4 decades old but is still useful


Software
Hardware
Application
Model / database
Application
program
Graphics
Library
Graphics
System/
GPU

Graphics Library - I

- Examples: OpenGL™, DirectX™, Windows Presentation Foundation™ (WPF), RenderMan™, HTML5+WebGL
- Primitives (characters, lines, polygons, meshes,...)
- Attributes
  - Color, line style, material properties for 3D

Graphics Library - II

- Lights
- Transformations
- Immediate mode vs. retained mode
    - **immediate mode**: no stored representation, package holds only attribute state, and application must completely draw each frame
    - **retained mode**: library compiles and displays from scenegraph that it maintains, a complex DAG. It is a display-centered extract of the Application Model

Microsoft®
DirectX® 11

PIXAR's RenderMan

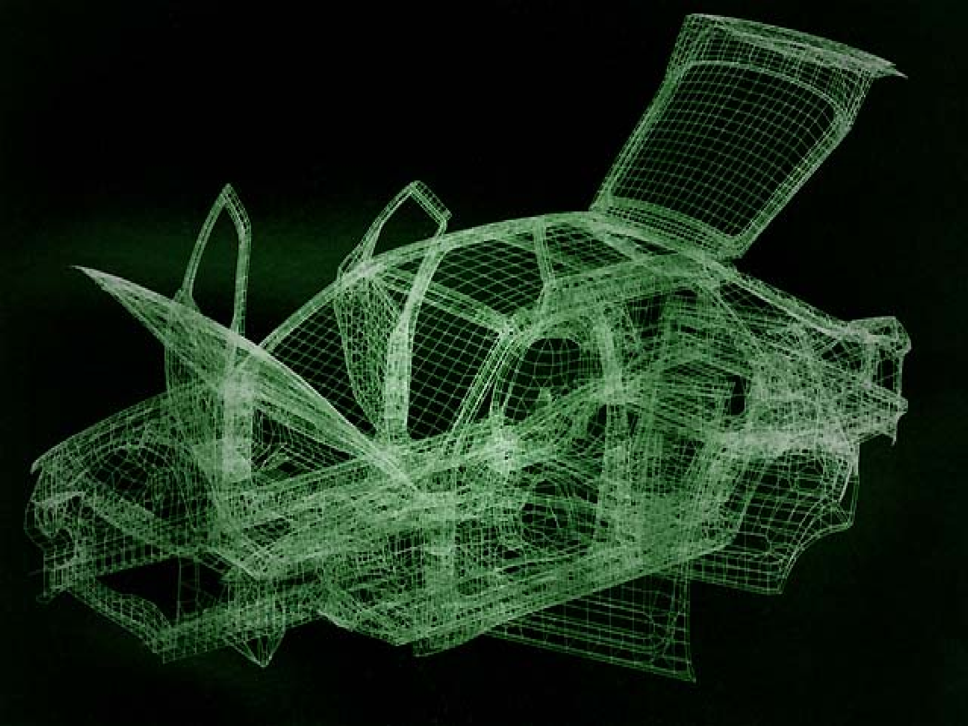
Sample-based Graphics

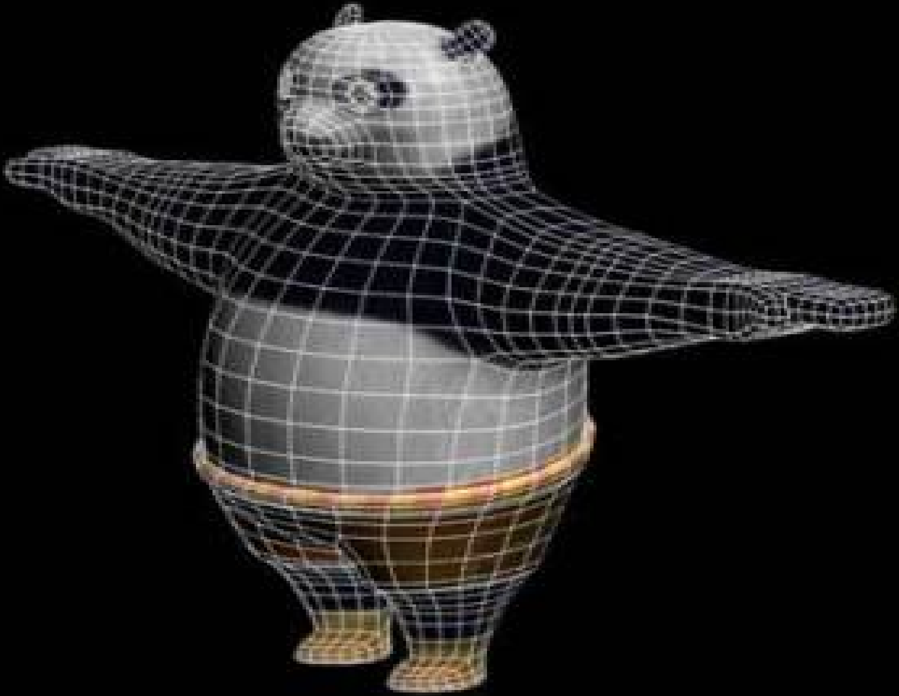
- Sample-based graphics: discrete samples are used to describe visual information
  - pixels can be created by digitizing images, using a sample-based "painting" program, etc.
  - often some aspect of the physical world is sampled for visualization, e.g., temperature across the US
  - example programs: Adobe Photoshop™, GIMP™ , Adobe AfterEffects™
  - Figure 2 presented at page 32 presents an example of Sample-based Graphics. You can clearly and easily notice the distortion and loss of data

Figure: Sample-based Graphic for a Building

Geometry-based Graphics

- Geometry-based graphics (also called scalable vector graphics or object-oriented graphics) : geometrical model is created, along with various appearance attributes, and is then sampled for visualization (rendering a.k.a image synthesis)
  - often some aspect of physical world is visually simulated, or "synthesized"
  - examples of 2D apps: Adobe Illustrator™, Adobe Freehand™, Corel CorelDRAW™
  - examples of 3D apps: Autodesk's AutoCAD™, Autodesk's (formerly Alias—Wavefront's) Maya™, Autodesk's 3D Studio Max™
  - Geometry-based Graphics models can be animated later.



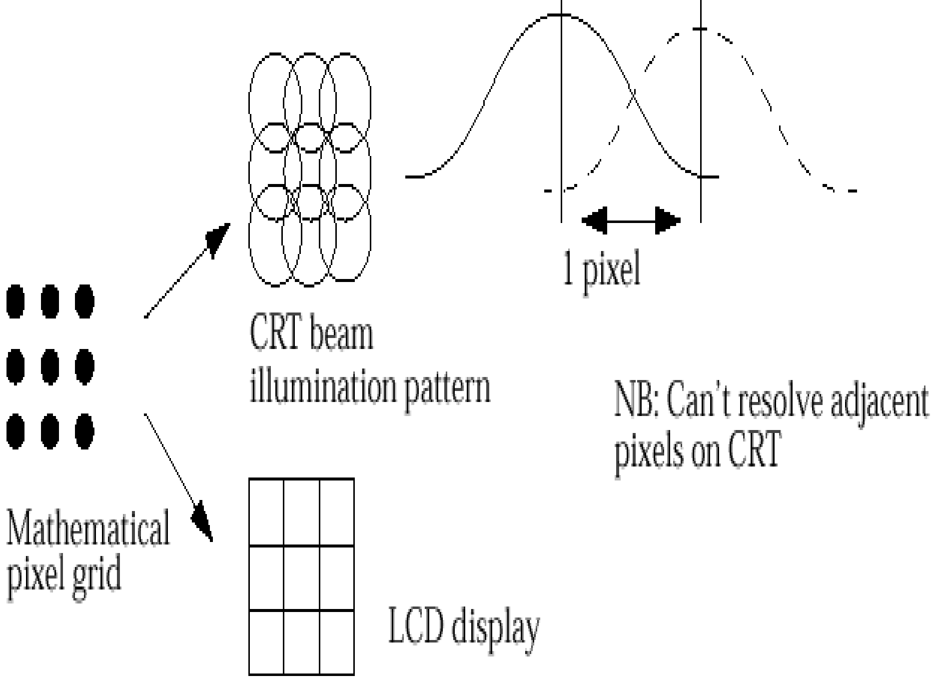


Sample-based Graphics - I

- Images are made up of grids of discrete pixels, or 2D "picture elements"
- **Pixels** are point locations with associated sample values, usually of light intensities/colors, transparency, and other control information
- When we sample an image, we sample the point location along the continuous signal and we cannot treat the pixels as little circles or squares
- Figure **??** presented at page **??** presents a comparison between CRT and LCD image manipulation of pixels
- Samples created directly in paint-type program, or as sampling of continuous (analog) visual materials. E.g., photograph can be sampled (light intensity/color

Sample-based Graphics - II

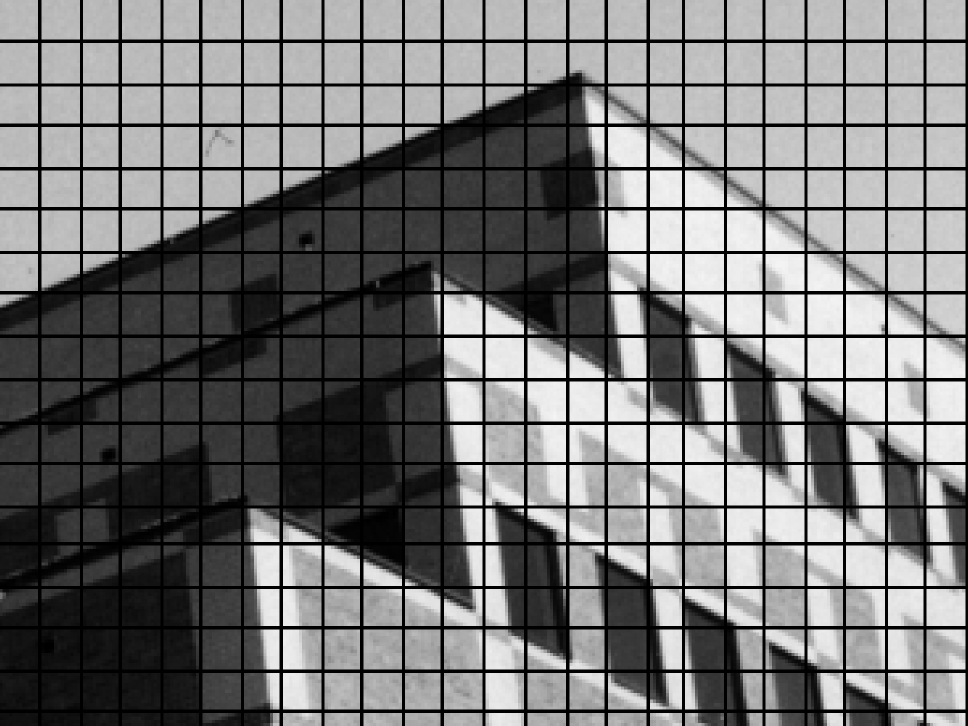
- Sample values can also be input numerically (e.g., with numbers from computed dataset)
- Once an image is defined as pixel-array, it can be manipulated
  - **Image editing**: changes made by user, such as cutting and pasting sections, brush-type tools, and processing selected areas
  - **Image processing**: algorithmic operations that are performed on image (or pre-selected portion of image) without user intervention. Blurring, sharpening, edge-detection, color balancing, rotating, warping.



1 pixel
CRT beam
illumination pattern
NB: Can't resolve adjacent
pixels on CRT
Mathematical
pixel grid
LCD display

Sampling an Image

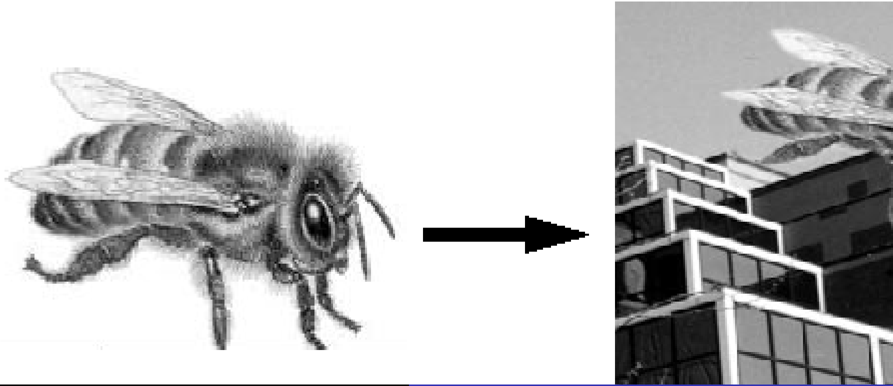
- Lets do some sampling of a building
- Figure **??** presented at page **??** presents a 3D Scene of the building
- Figure **??** presented at page **??** presents the input building image that will be sampled
- A color value is measured at every grid point and used to color corresponding grid square. Used measurements are: $0 =$ white, $5 =$ gray, $10 =$ black. Figure **??** presented at page **??** presents the proposed equivalent grid
- Poor sampling and image reconstruction method creates blocky image, the one presented at figure **??** presented at page **??**



Advantages of Sampling Images

- Once image is defined in terms of colors at (x, y) locations on grid, can change image easily by altering location or color values
- E.g., if we reverse our mapping above and make 10 = white and 0 = black, the image would look like this:
- Pixel information from one image can be copied and pasted into another, replacing or combining with previously stored pixels
- Figure 3 presented at page 45 presents clearly the main advantage of sampling images

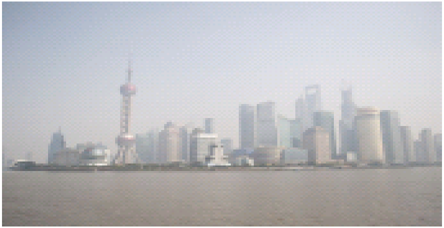
Sampling Image's Main Advantage



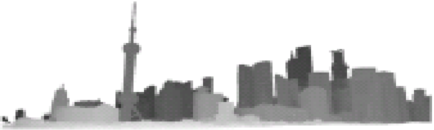
Sampling Images Disadvantages

WYSIAYG (What You See Is All You Get): No additional information

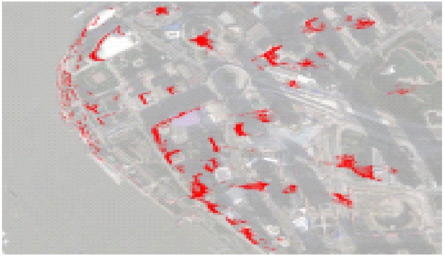
- no depth information
- can't examine scene from different point of view
- at most can play with the individual pixels or groups of pixels to change colors, enhance contrast, find edges, etc.
- But recently, strong interest in image-based rendering to fake 3D scenes and arbitrary camera positions. New images constructed by interpolation, composition, warping and other operations.
- meaning of no-depth information that is missing from sampling images, by presenting a reconstruction method through depth information for two images taken from



(a) One input image



(b) Depth map



(c) Map overlay

(d) Rendering

Results on a challenging unstructured light field, obtained by hand-held capture (a) from a floating boat. (b) A resulting depth map. (c) Overlay of our reconstruction on a satellite image ©2013 DigitalGlobe, Google. (d) Rendering from a novel viewpoint.

Geometry-Based Graphics I

- Geometry-based graphics applications store mathematical descriptions, or "models," of geometric elements (lines, polygons, polyhedrons. . .) and associated attributes (e.g., color, material properties). Elements are primitive geometric shapes, primitives for short
- Images created as pixel arrays (via sampling of geometry) for viewing, but not stored as part of model. Images of many different views are generated from same model

Geometry-Based Graphics II

- Users cannot usually work directly with individual pixels in geometry-based programs; as user manipulates geometric elements, program resamples and redisplays elements
- Increasingly rendering combines geometric and sample-based graphics, both as performance hack and to increase quality of final product

Geometric Modeling - I

- What is a model?
- Captures salient features (data, behavior) of thing/phenomenon being modeled
    - data includes geometry, appearance, attributes...
    - note similarity to OOP ideas
- Real: some geometry inherent
    - physical (e.g., actual object such as a chair)
    - non-physical (e.g., mathematical function, weather data)

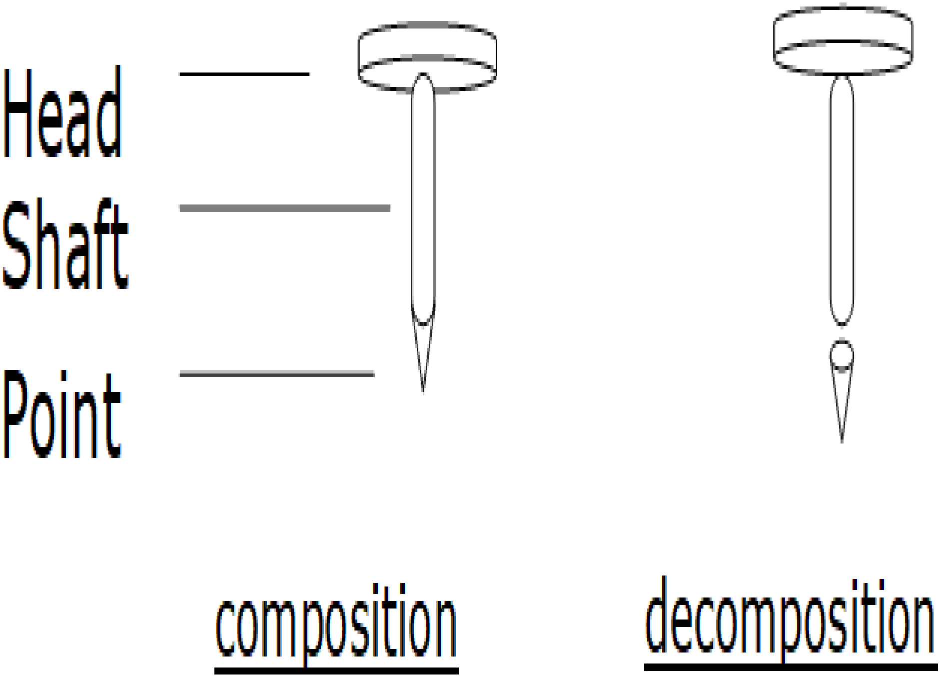
Geometric Modeling - II

- Abstract: no inherent geometry, but for visualization
  - organizational (e.g., company org. chart)
  - quantitative (e.g., graph of stock market)
- Modeling is coping with complexity
- Our focus: modeling and viewing simple everyday objects
- Consider this: Through 3D computer graphics, first time in human history we have abstract, easily changeable 3D forms. This has revolutionized working process of many fields – science, engineering, industrial design, architecture, commerce, entertainment, etc. This has profound implications for visual thinking and visual literacy. . .

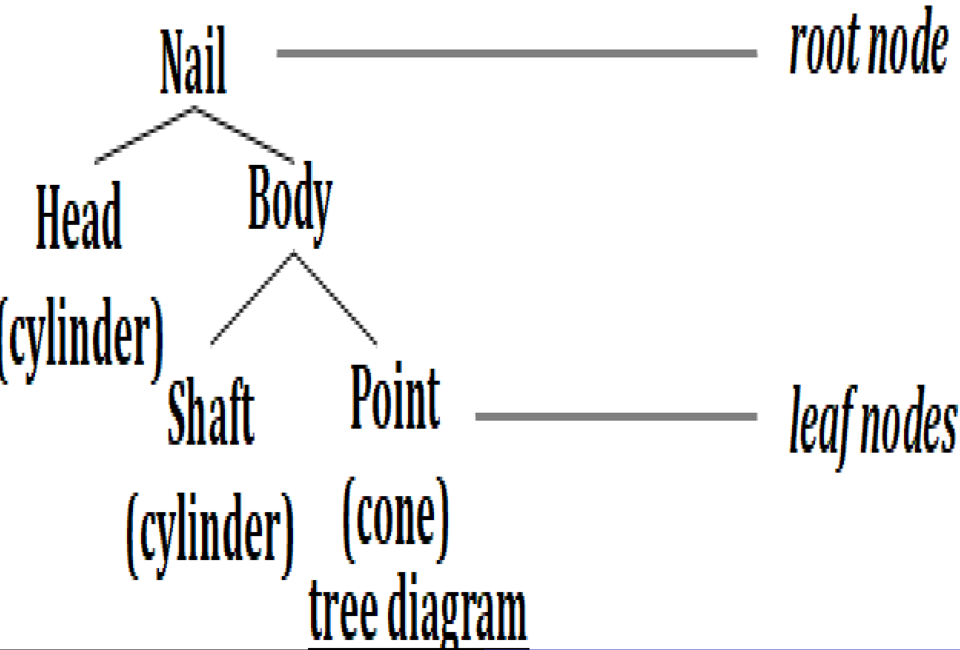

Point Light
Spot Light
Directional Light
Ambient Light

Decomposition of Geometric Model

- Divide and Conquer
- Hierarchy of geometrical components
- Reduction to primitives (e.g., spheres, cubes, etc.)
- item Simple vs. not-so-simple elements (nail vs. screw)
- Figure **??** at page **??** presents the idea of Composition and Decomposition of Geometric Models

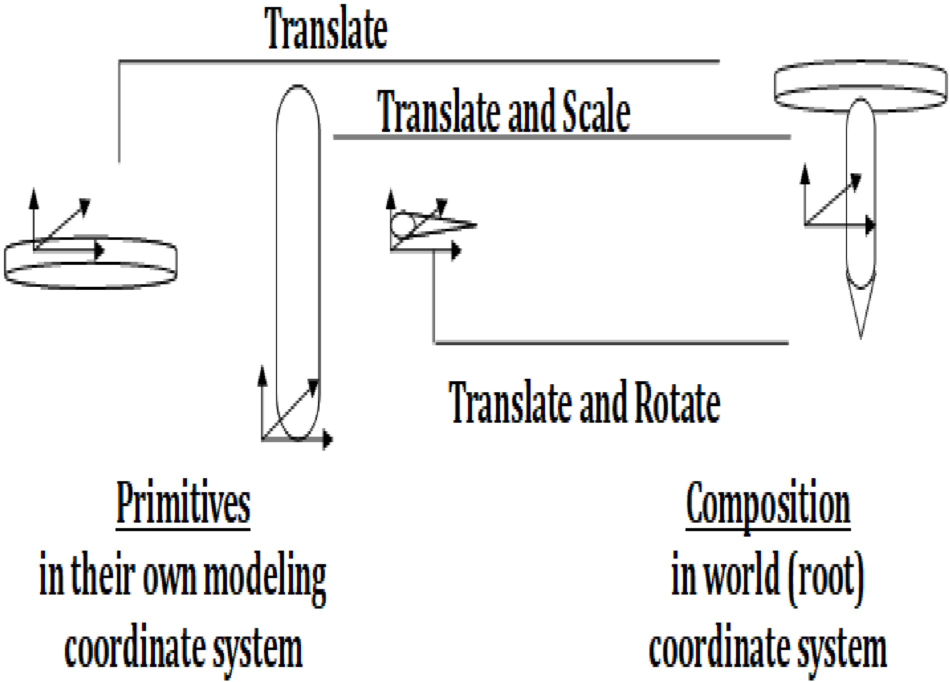

Head
Shaft
Point
composition
decomposition

- Object to be modeled is (visually) analyzed, and then decomposed into collections of primitive shapes.
- Tree diagram provides visual method of expressing "composed of" relationships of model
- Such diagrams are part of 3D program interfaces (e.g., 3D Studio MAX, Maya)
- As a data structure to be rendered, it is called a scenegraph
- Figure **??** presented at page **??** presents a Hierarchical (Tree) Diagram of Nail. This Tree hierarchy can be used in modeling the Nail


Nail
root node
Head
(cylinder)
Body
Shaft
(cylinder)
Point
(cone)
leaf nodes
tree diagram

Composition of Geometric Model

- Figure **??** presented at page **??** presents the Primitives created in decomposition models
- We can mix those primitives to manipulate the output of the generated model
- Primitives created in decomposition process must be assembled to create final object. Done with affine transformations, T, R, S (as in above example).


Translate
Translate and Scale
Translate and Rotate
Primitives
in their own modeling
coordinate system
Composition
in world (root)
coordinate system