Computer Graphics

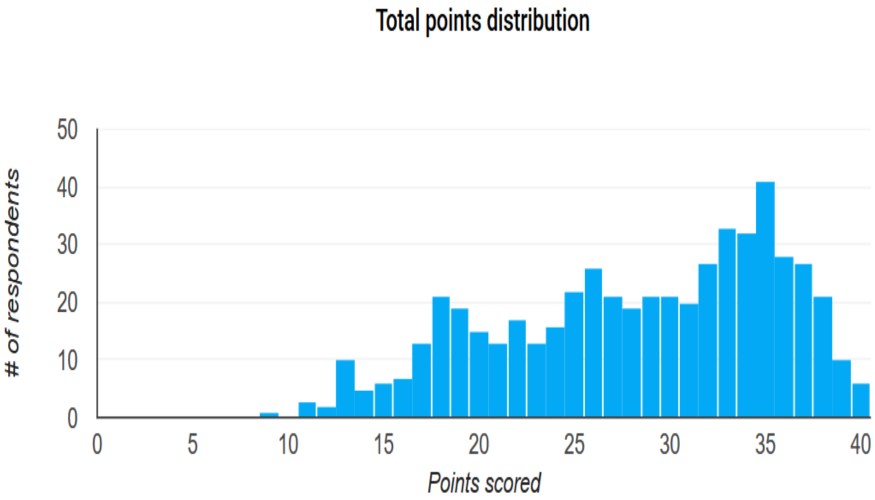
Lecture - 03

Haitham A. El-Ghareeb

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

February 23, 2019

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



  - 3D Graphics
- Strategy Games
- Simulation Games

Character Display

- Character Displays (1960s – now) - Figure 1 at page 7
- 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

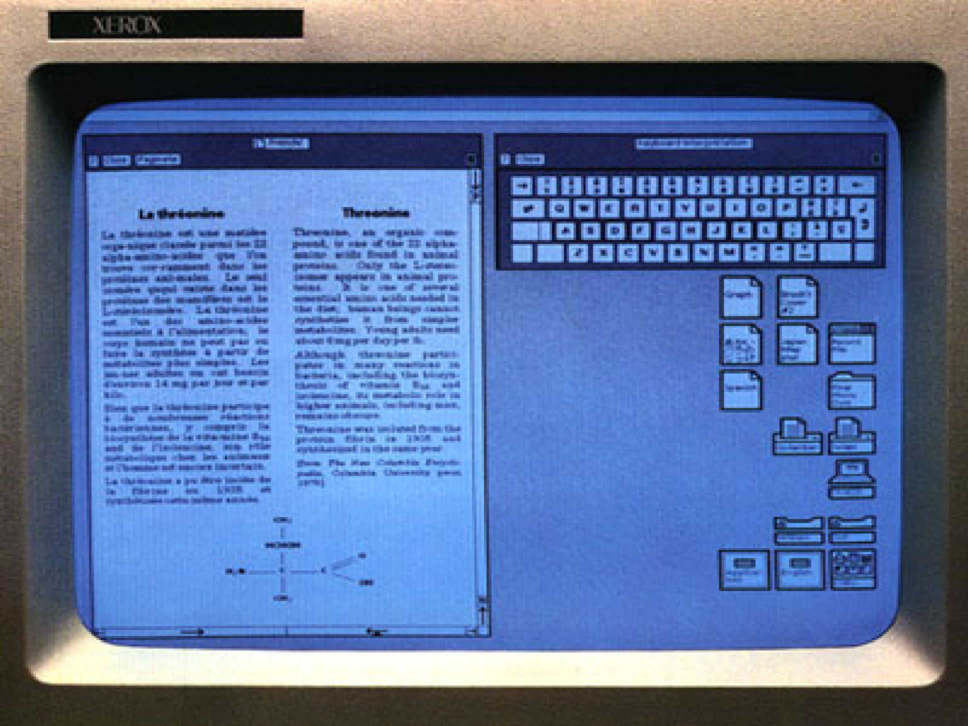
- Vector (Calligraphic, Line Drawing) - Figure **??** at page **??**
- Displays (1963 – 1980s)
- Display: line drawings and stroke text; 2D and 3D transformation hardware
- Object and command specification: command-line typing, function keys, menus
- Control over appearance: pseudo-WYSIWYG
- Application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

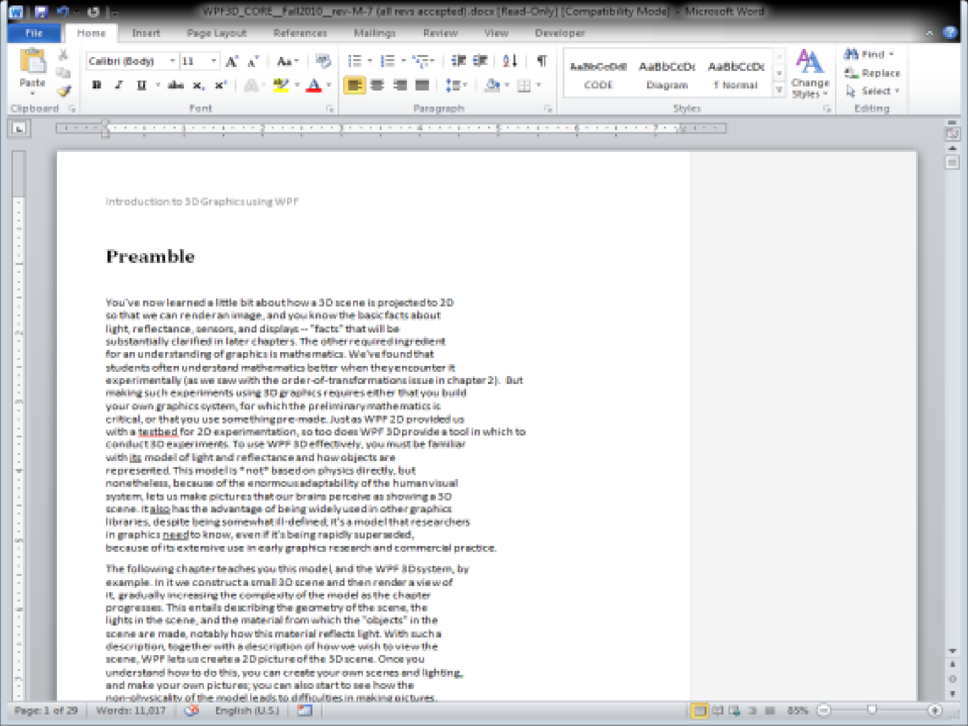
2D bitmap - I

- 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

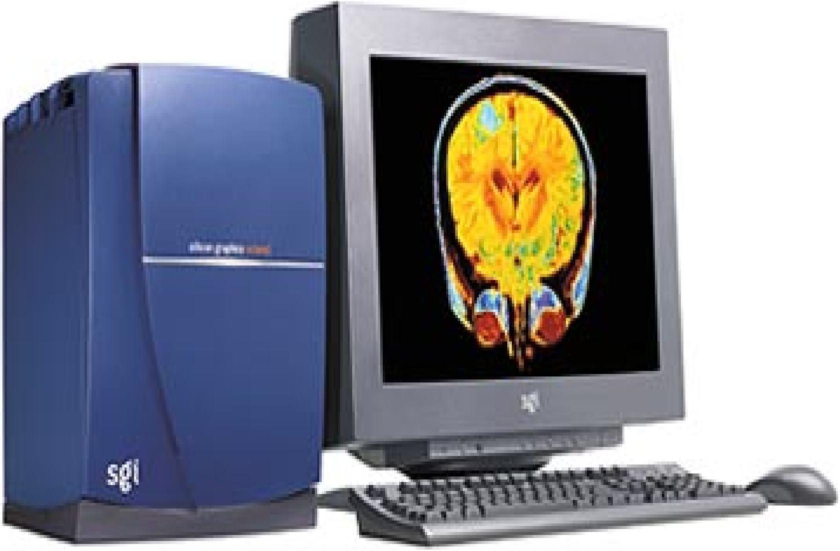
You've now learned a little bit about how a 3D scene is projected to 2D so that we can render an image, and you know the basic facts about light, reflectance, sensors, and displays -- "facts" that will be substantially clarified in later chapters. The other required ingredient for an understanding of 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

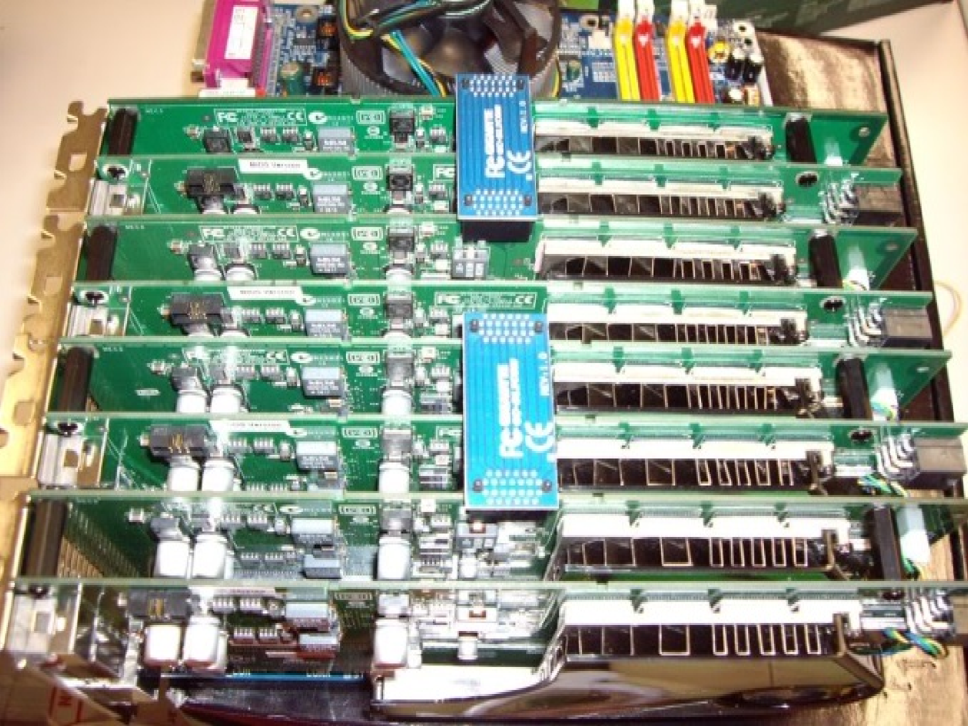
- 3D graphics workstations (1984 at SGI – now)
- Display: real-time, pseudo-realistic images of 3D scenes
- Object and command specification: 2D, 3D and N-D input devices (controlling 3+ degrees of freedom) and force feedback haptic devices for point-and-click, widgets, and direct manipulation
- Control 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™



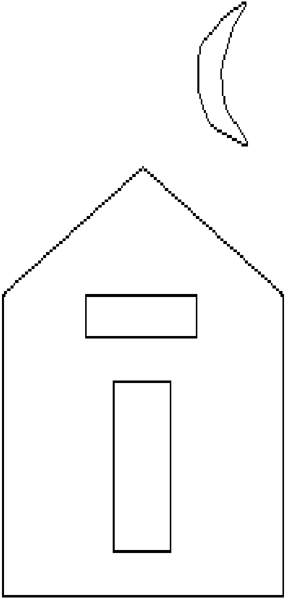
High-end PCs

- 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

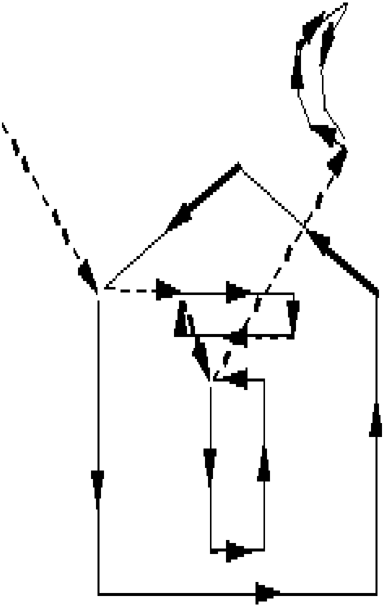


Vector

- 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



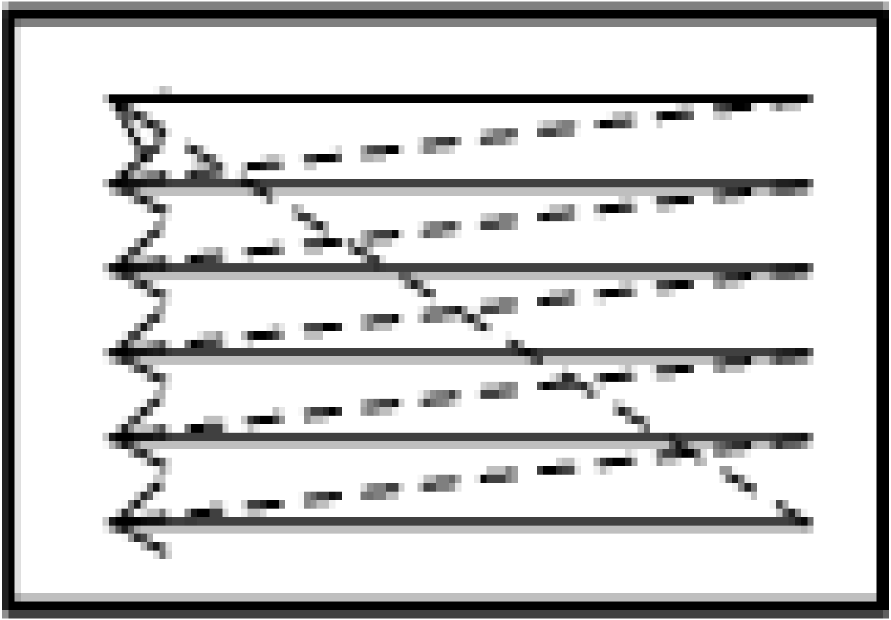
Ideal Drawing

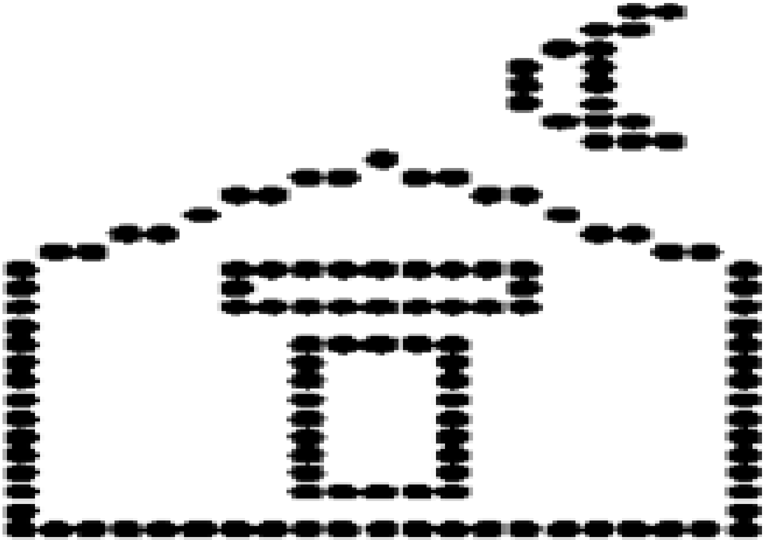


Vector Drawing

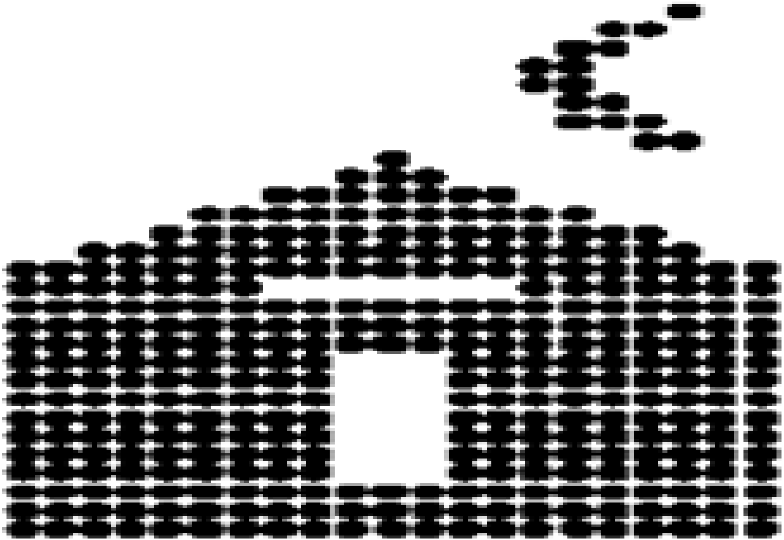
Raster

- (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





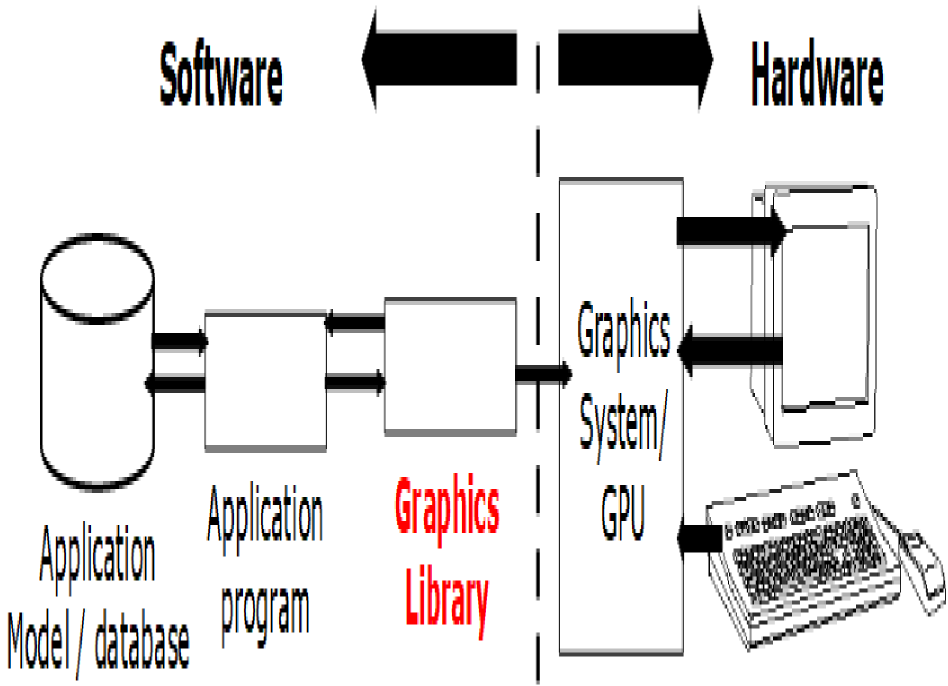
outline primitives



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

OpenGL™

Microsoft® DirectX® 11

PIXAR's
RenderMan

HTML

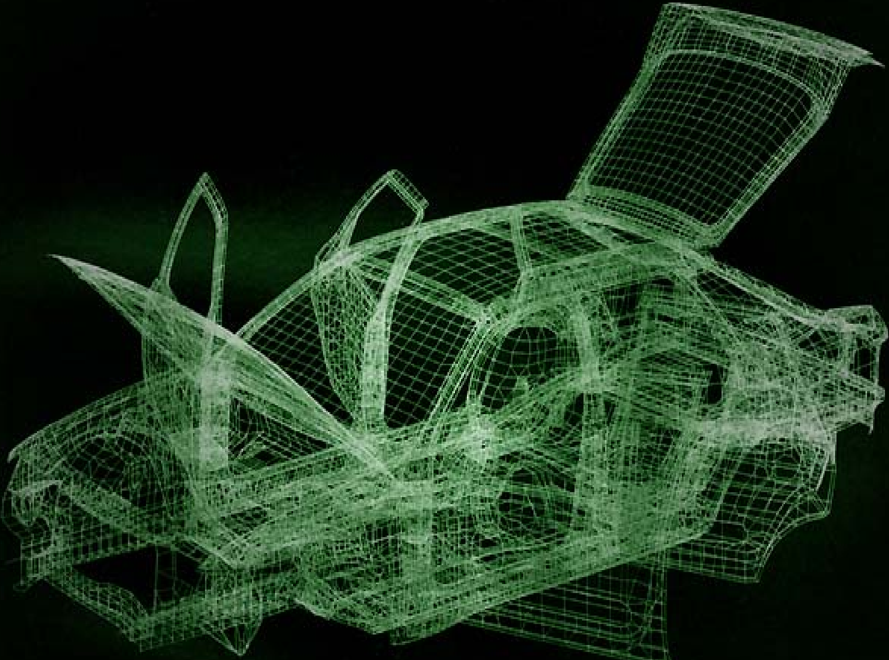
Sample-based Graphics

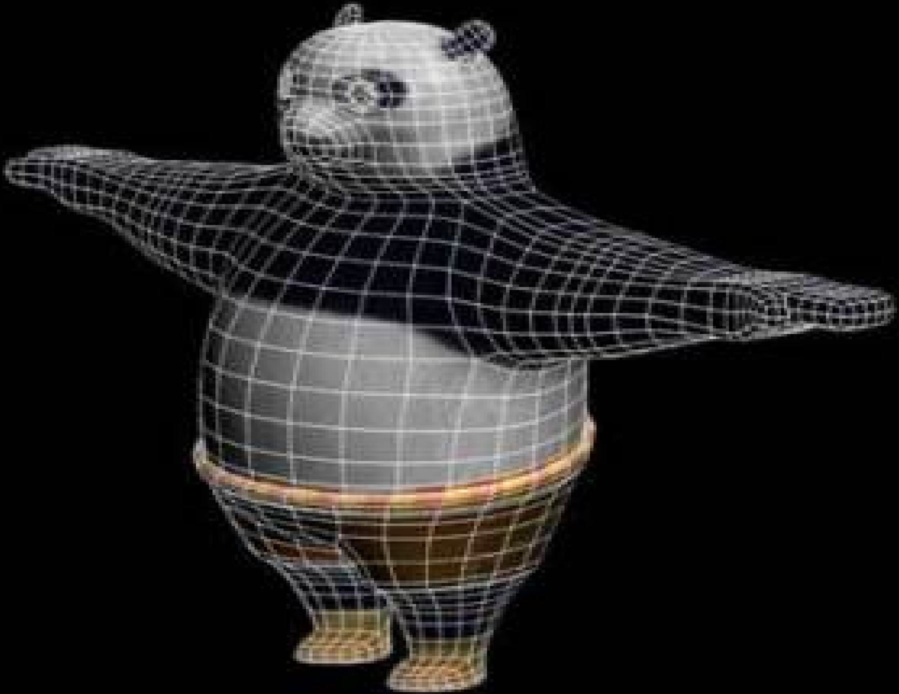
- Sample-based graphics: discrete samples are used to describe visual information
  - pixels can be created by digitizing images, using a sample-based “painting” program, etc.
  - often some aspect of the physical world is sampled for visualization, e.g., temperature across the US
  - example programs: Adobe Photoshop™, GIMP™ , Adobe AfterEffects™
  - Figure 2 presented at page 33 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.



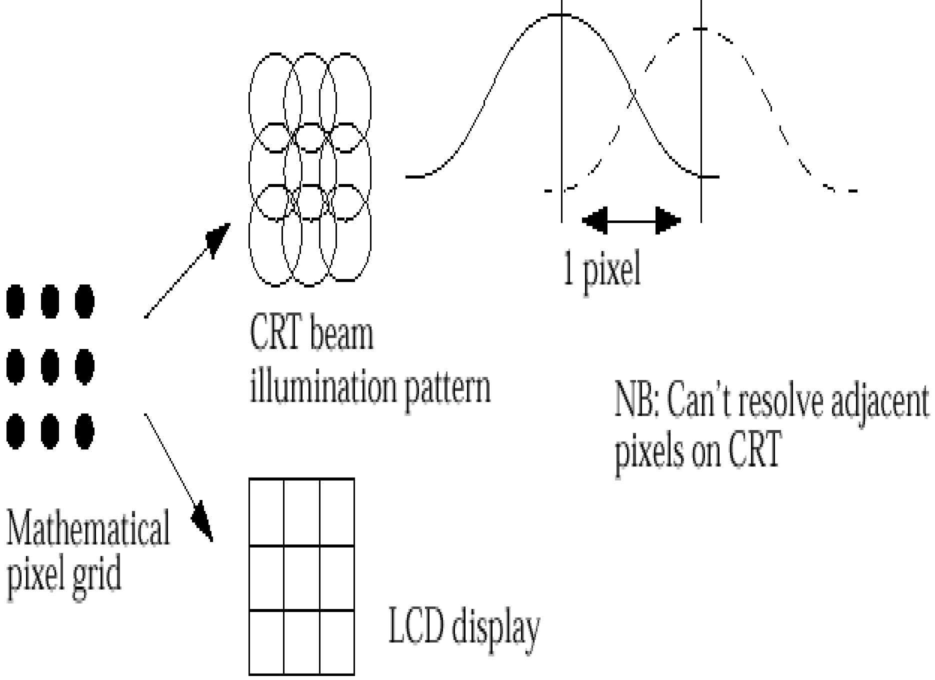


Sample-based Graphics - I

- 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

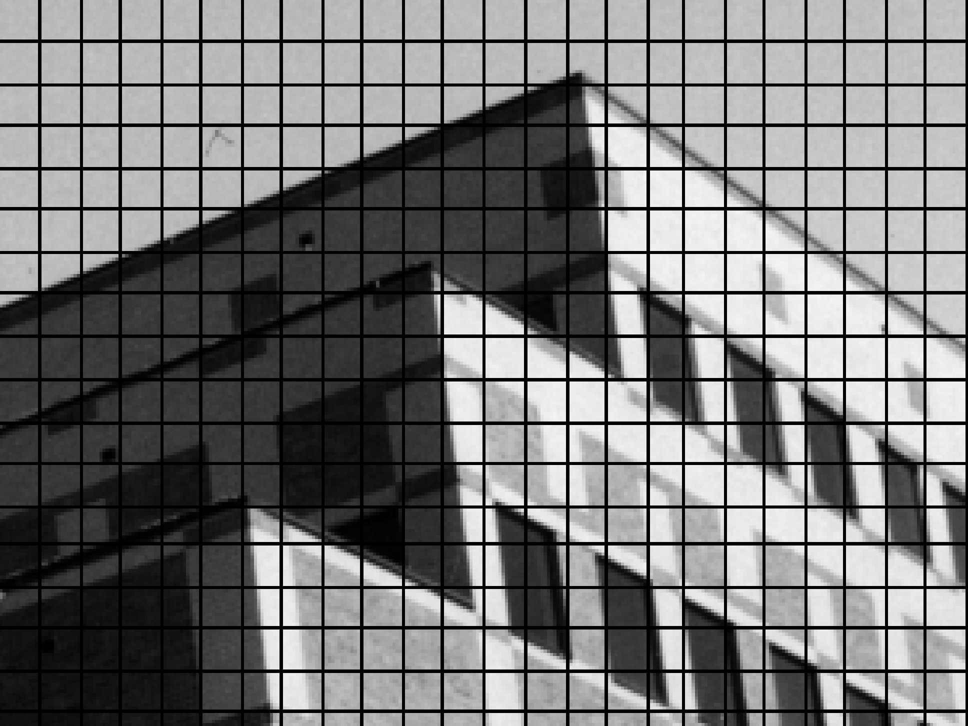
Sample-based Graphics - II

- 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

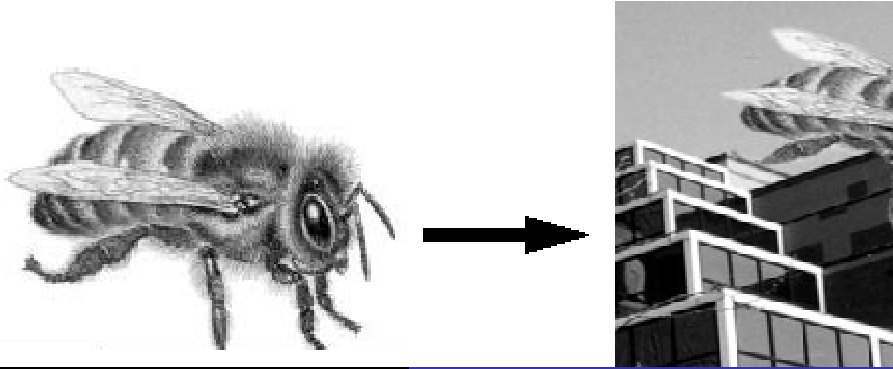
- Lets do some sampling of a building
- Figure **??** presented at page **??** presents a 3D Scene of the building
- Figure **??** presented at page **??** presents the input building image that will be sampled
- A color value is measured at every grid point and used to color corresponding grid square. 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 46 presents clearly the main advantage of sampling images

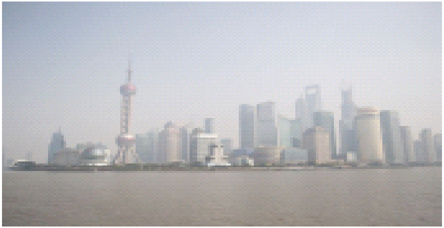
Sampling Image's Main Advantage



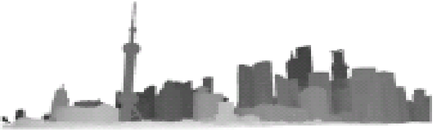
Sampling Images Disadvantages

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

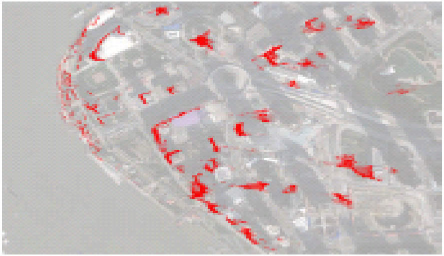
- no depth information
- can't examine scene from different point of view
- at most can play with the individual pixels or groups of pixels to change colors, enhance contrast, find edges, etc.
- But recently, strong interest in image-based rendering to fake 3D scenes and arbitrary camera positions. New 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



(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)

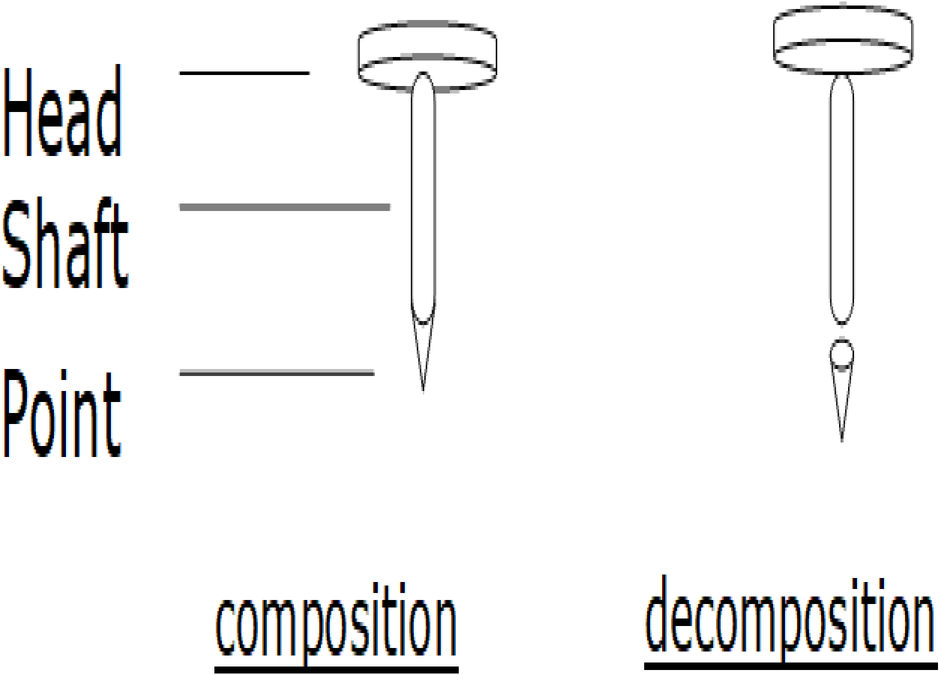
Geometric Modeling - II

- 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. . .

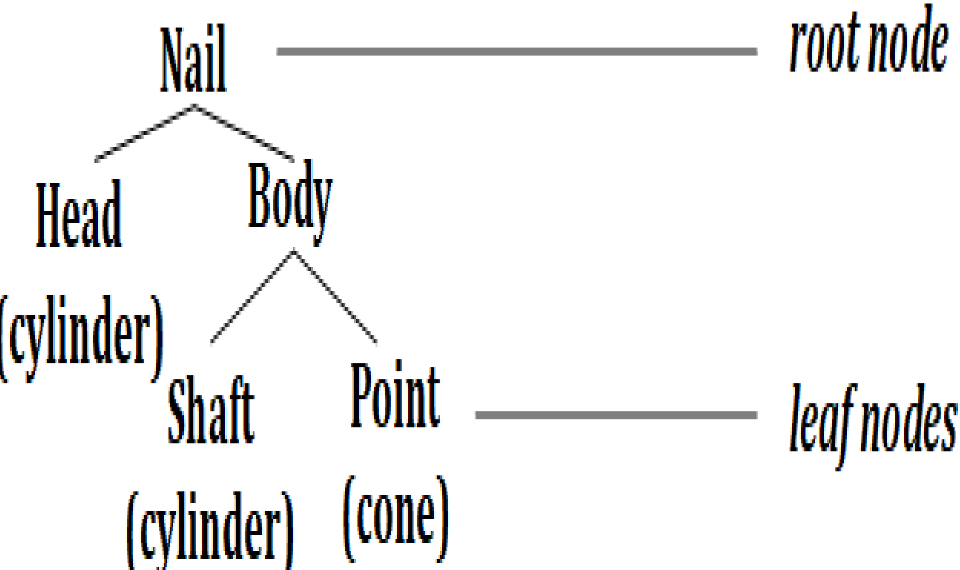

Point Light
Spot Light
Directional Light
Ambient Light

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

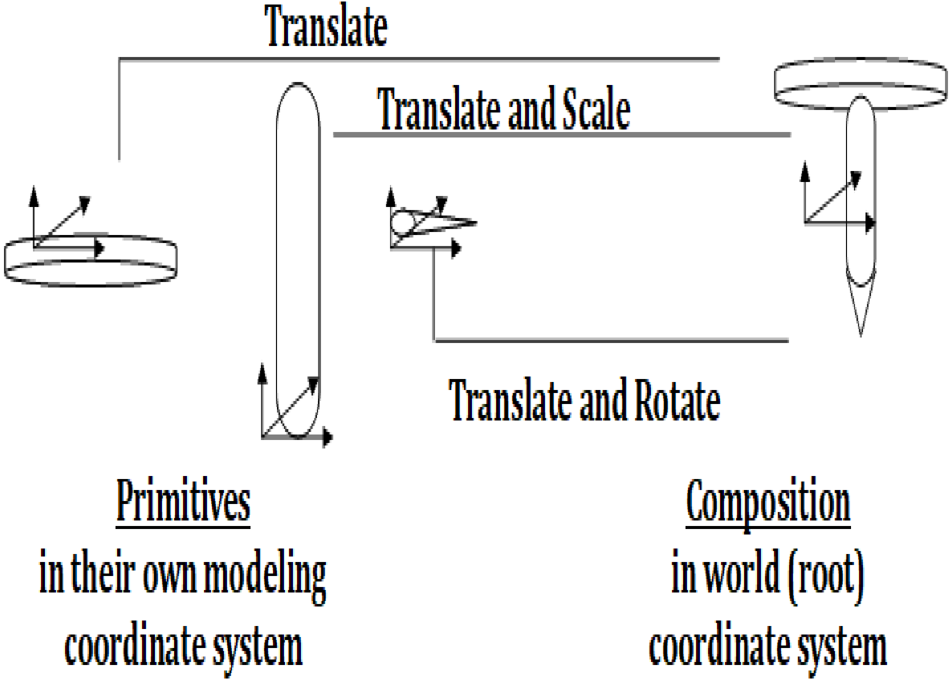
- Object to be modeled is (visually) analyzed, and then decomposed into collections of primitive shapes.
- Tree diagram provides visual method of expressing "composed of" relationships of model
- Such diagrams are part of 3D program interfaces (e.g., 3D Studio MAX, Maya)
- As a data structure to be rendered, 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



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

Gentle Introduction

- No Mathematics

Gentle Introduction

- No Mathematics
- in this Lecture

Different Types of Games

- First Person Shooter (FPS)
- Strategy Games
- Simulation Games

First Person Shooter (FPS)

- Type of game where you run around 3D levels carrying a big gun shooting stuff
- Examples of this sort of game include Doom , Quake , Half Life , Unreal or Goldeneye
- There are other games that look very similar, but aren't first person shooters, for instance Zelda: Ocarina of Time or Mario 64

Strategy Games

- The Strategy games are divided into two main types
    - Real Time Strategy (RTS)
    - Turn Based Strategy
- These games usually involve building and managing a city or civilization and also fighting wars by controlling troops
- Examples of real time strategy games are Age of Empires , Command and; Conquer , Tiberian Sun
- Examples of turn based strategy games are Civilization and Alpha Centauri

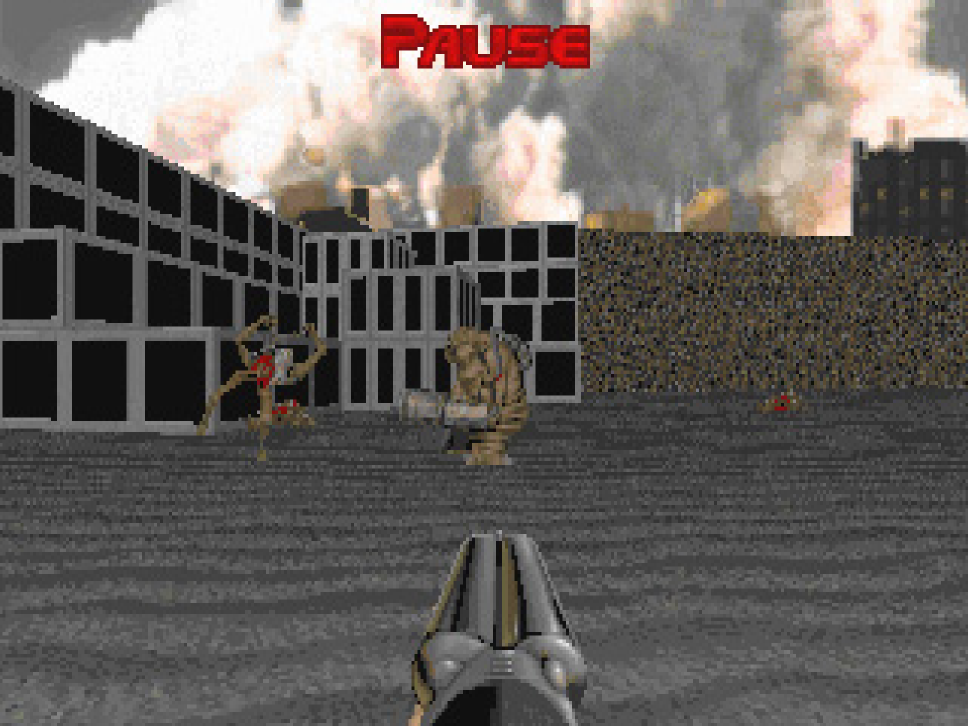
Simulation Games

- Games that try to make something as realistic as possible
- For instance, Flight Sims are computer games which try to realistically simulate flying an aeroplane or helicopter
- Two games of this sort are Microsoft Flight Simulator and Red Baron
- Space sims are like flight sims, but with spaceships instead of planes.
- For instance, Wing Commander or X-Wing vs. Tie Fighter .
- Racing games are games which simulate driving different sort of cars.
- For instance, Need for Speed , NASCAR Racing , Gran

First Person Shooters

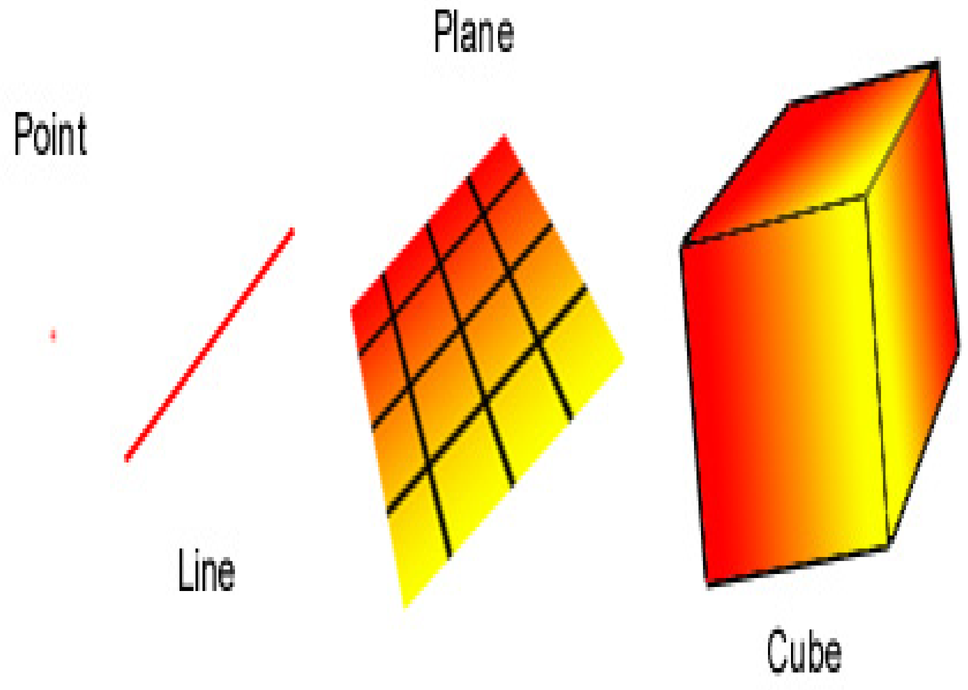
- The most amazing things about FPS are their incredible graphics.
- They look almost real, none of this would have been possible without the use of advanced maths.
- Here are some pictures from the early games (Wolfenstein) to the most recent games (Quake III Arena).
- All of the following screen shots are from games by iD software.


FLOOR
6
SCORE
0
LIVES
1
HEALTH
32%
AMMO
99


PAUSE

Geometry, Vectors, and Transformations

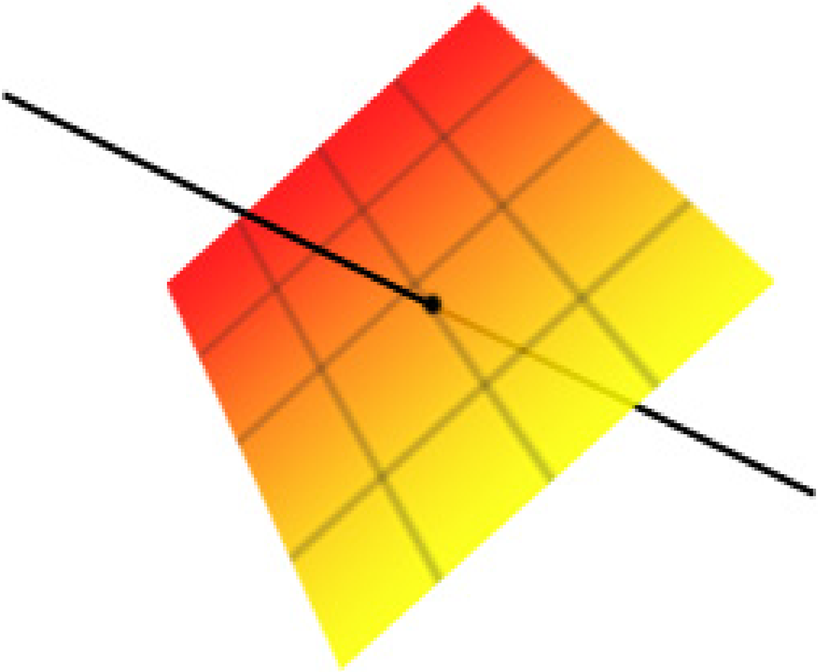
- Geometry is the study of shapes of various sort
- The simplest shape is the point
- It's quite difficult to explain what a point is, it is basically just a position
- Another simple shape is a straight line
- A straight line is just the simplest shape joining two points together
- A plane is a more complicated shape, it is a flat sheet, like a piece of paper or a wall
- There are more complicated shapes, called solids , like a cube or a sphere



Plane
Point
Line
Cube

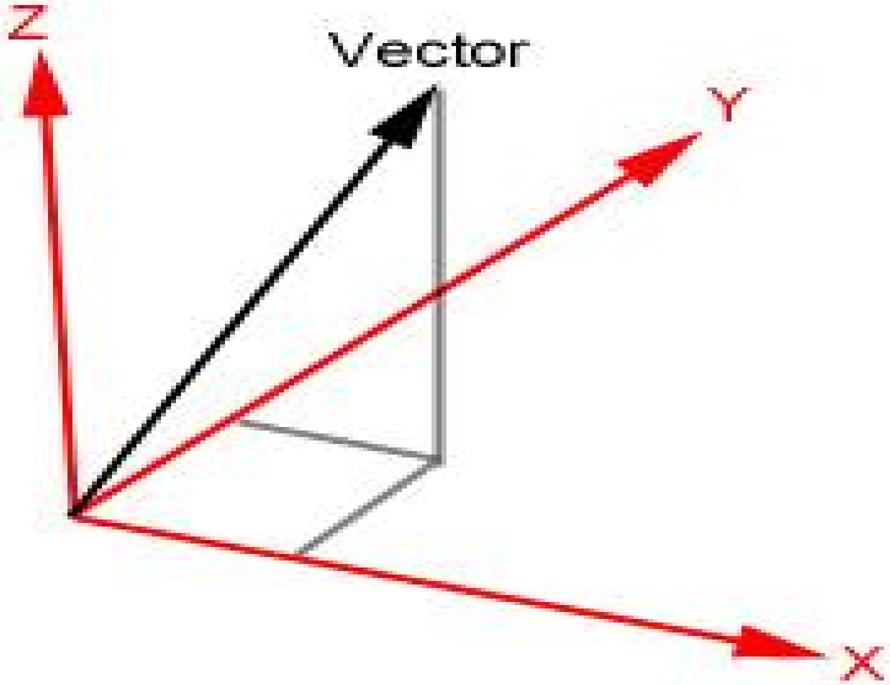
Intersection of a Line and Plane

- If you have a line and a plane, you can find the point where the line cuts through the plane
- In fact, sometimes you can't find the intersection, because they don't meet and sometimes the line is inside the plane so they meet at every point on the line
- We call this the intersection of the line and the plane



Vector

- A vector is a mathematical way of representing a point
- A vector is 3 numbers, usually called x, y and z
- You can think of these numbers as how far you have to go in 3 different directions to get to a point
- For instance, put one arm out pointing to the right, and the other pointing straight forward
- I can now give you a vector and you'll be able to find the point I'm talking about
- For instance, if I say x=3, y=1, z=5, you find the point by walking 3 metres in the direction of your right hand, then 1 metre in the direction of your left hand, and then getting a ladder and climbing up 5 metres

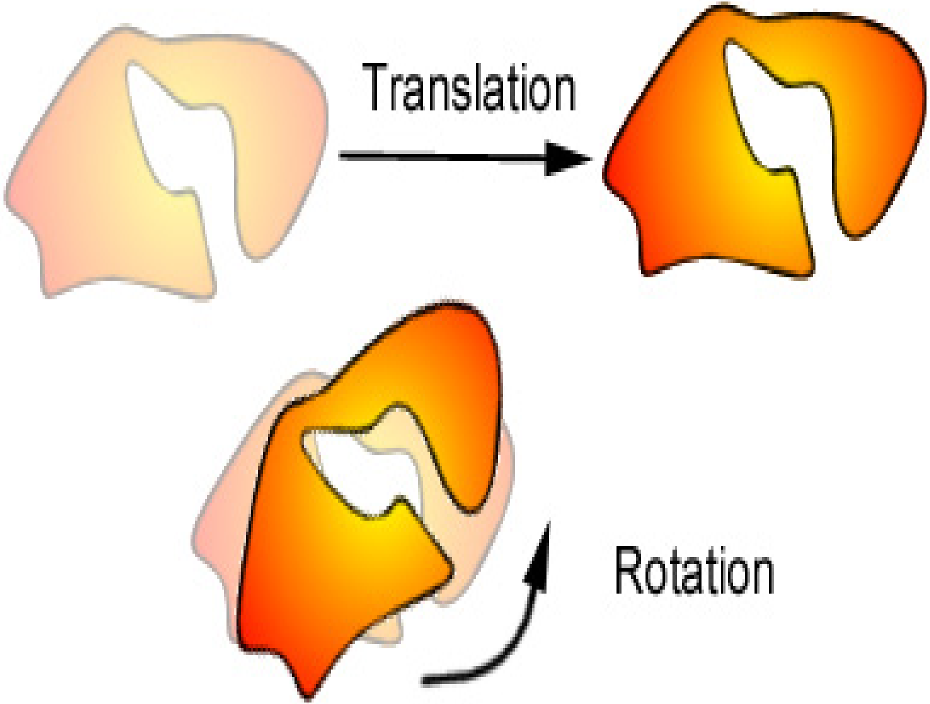

Z
Vector
Y
X

Confusing Part about Vectors

- Vectors are written as (x,y,z), for instance (1,2,3) means move 1 in the x-direction, 2 in the y-direction and 3 in the z-direction.
- One confusing thing about vectors is that they are sometimes used to represent a point, and sometimes they are used to represent a direction.
- The vector (1,0,0) can mean both "the point you get to if you move 1 unit in the x-direction from the starting point", or it can mean "move 1 unit in the x-direction from where you are now".

Transformation

- A transformation moves a point (or an object, or even an entire world) from one place to another
- For instance, I could move it to the right by 4 metres, this type of transformation is called a translation
- Another type of transformation is rotation
- If you take hold of an object (a pen for instance), and twist your wrist, you have rotated that object


Translation
Rotation

3D Graphics I

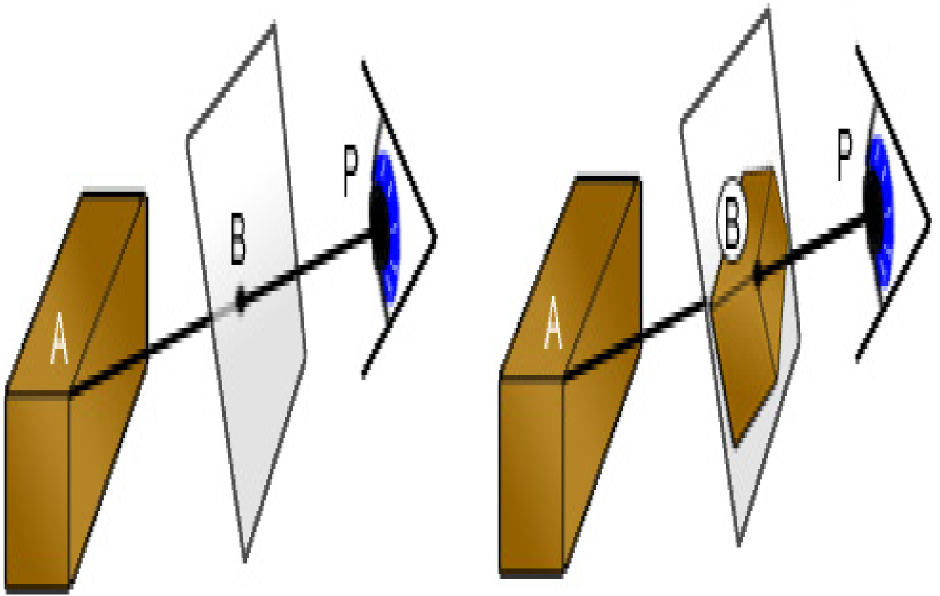
- The basic idea of 3D graphics is to turn a mathematical description of a world into a picture of what that world would look like to someone inside the world
- The mathematical description could be in the form of a list, for instance
  - there is a box with centre (2,4,7) and sides of length 3
  - the colour of the box is a bluish grey
- To turn this into a picture, we also need to
  - describe where the person is
  - what direction they are looking
  - for instance: there is a person at (10,10,10) looking directly at the centre of the box
- From this we can construct what the world would look

3D Graphics II

- Imagine there is a painter whose eyes are at the point P.
- Imagine that he has a glass sheet which he is about to paint on.
- In the room he is painting, there is a wooden chest.
- One of the corners of the chest is at point A, and the painter wants to know where that corner of the chest should be on his glass sheet.
- The way he works it out is to draw a line L from his eyes (P) to the corner of the chest (A), then he works out where this line goes through the canvas, B.

3D Graphics III

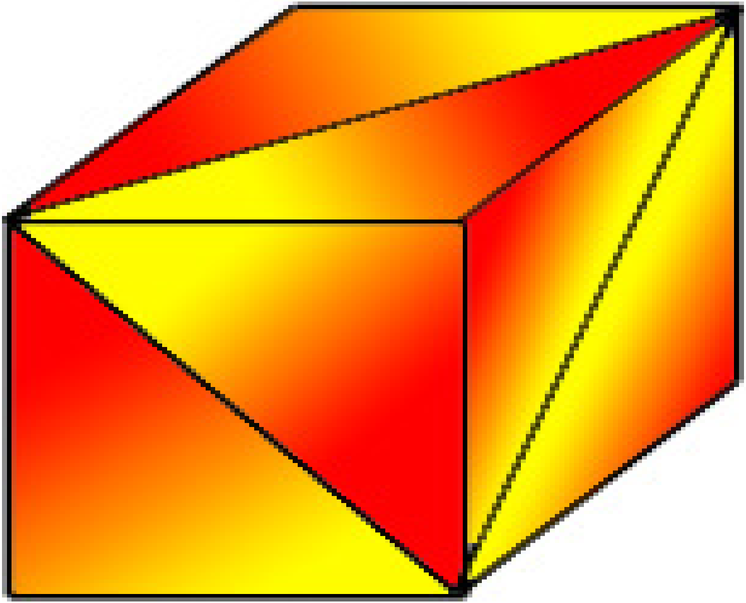
- He can do this, because the glass sheet is a plane, and I mentioned that you can find the intersection of a line and a plane above.
- This point B is where the corner of the chest should be in his painting.
- He follows this rule for every bit of the chest, and ends up with a picture which looks exactly like the chest.
- Here are two pictures, the first one shows the painting when he has only painted the one corner of the chest
- the second one shows what it looks like when he has painted the entire chest
- This process is called: Projection to a Plane



A
B
P
A
B
P

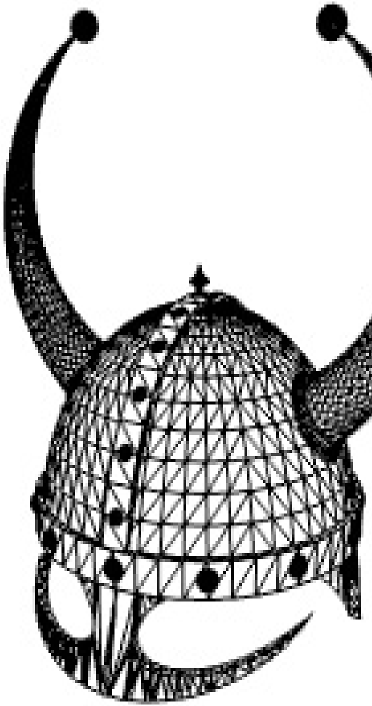
3D Graphics IV

- What I just described above is similar to what the computer is doing (50 times a second!)
- every time you run around shooting hideous monsters in Quake , although the details are slightly different.
- In computer games (at the moment) the description of the world is just a list of triangles and colours.
- The newest computer games are using more complicated descriptions of the world, using curved surfaces, NURBS and other strange sounding things, however in the end it always reduces to triangles.
- For instance, a box can be made using triangles as illustrated below.



3D Graphics V

- Here is a much more complicated example, using thousands of triangles.
- The first picture shows the triangles used
- the second picture is what it looks like with colours put in
- The reason for using triangles is that they are a very simple shape
- if you make sure that everything is made from only one type of shape, you don't have to write a separate program for each type of shape in the game.



3D Graphics VI

- Each time the computer draws a picture of the world, it goes through the following steps:
    - Firstly, it transforms the world (by rotating and translating), so that the person is at position (0,0,0) and the centre of the glass sheet (the centre of the screen in computers) is at (1,0,0). This makes the rest of the calculations much easier.
    - Secondly, it removes all the triangles you can't see so that it can forget about them, for instance the triangles that are behind you or the ones that are so far away that you can't see them.

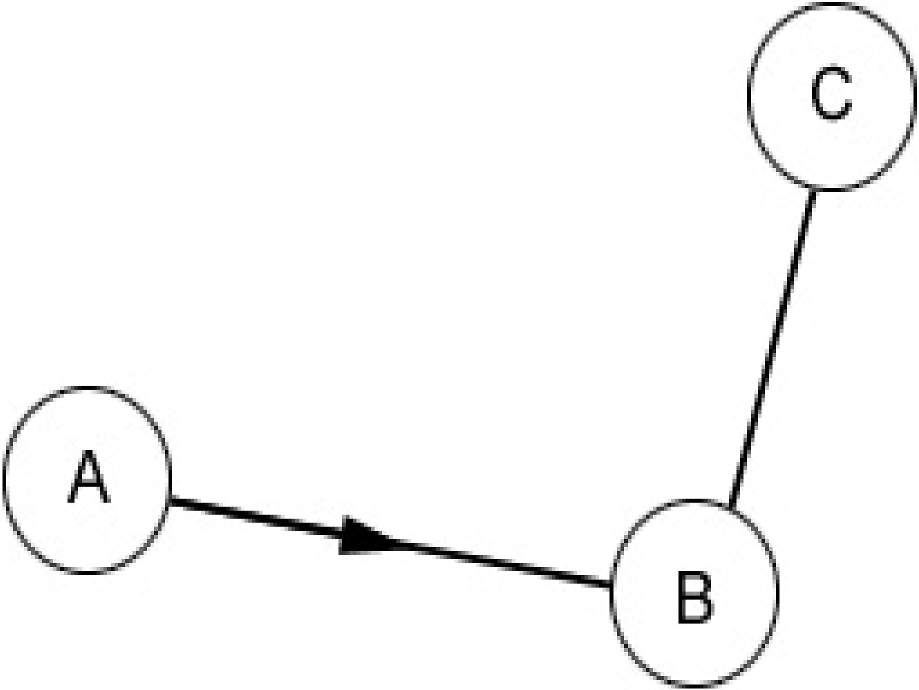
- Each time the computer draws a picture of the world, it goes through the following steps: (cont.)
  - Thirdly, for every remaining triangle, it works out what it would look like when painted on the glass sheet (or drawn on the screen in computers).
  - Finally, it puts the picture it has drawn on the screen.
- Nowadays, computers are so fast that they can draw hundreds of thousands of triangles every second, making the pictures more and more realistic

Strategy Games

- Strategy games usually have much simpler graphics than FPS games
- When you click on a little soldier in a strategy game, and then click somewhere else, telling him that he should walk to the place where you have clicked, what happens inside the computer?
- How does the computer know how to make the soldier get from where he already is to where he is going.
- So you can't just say, "look at the map and work out the best route to wherever you are going", he needs to be given exact instructions at every stage of his journey.

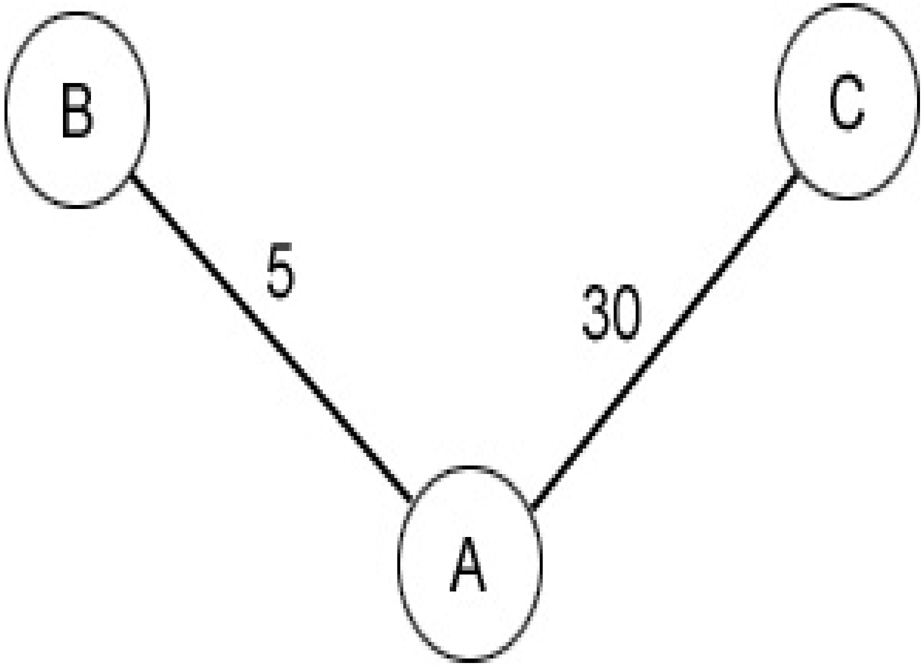
Nodes, Edges, and Graphs

- This problem is called path finding
- To explain how the computer works out the best route, you need to know what nodes , edges and graphs are.
- You may have heard of graphs before in maths, but they mean something slightly different here.
- The simplest example of nodes and graphs is a map of some cities, and the roads between them (or an underground map).
- Each city is a node, usually drawn as a circular blob.
- Each road is an edge, and connects two nodes (cities), these are usually drawn as straight lines.
- The whole collection of nodes and edges (cities and


C
A
B

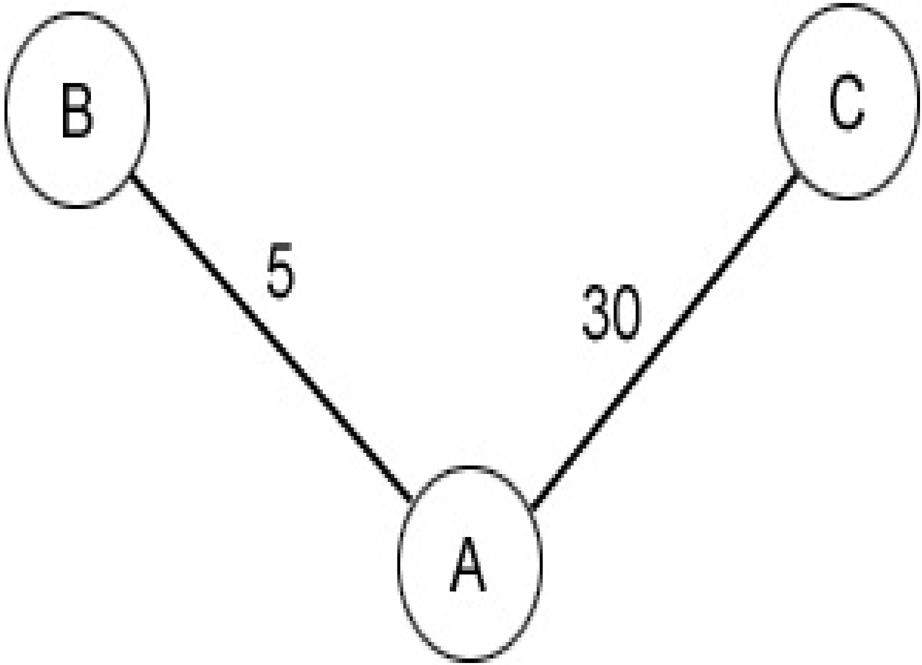
Nodes, Edges, and Graphs II

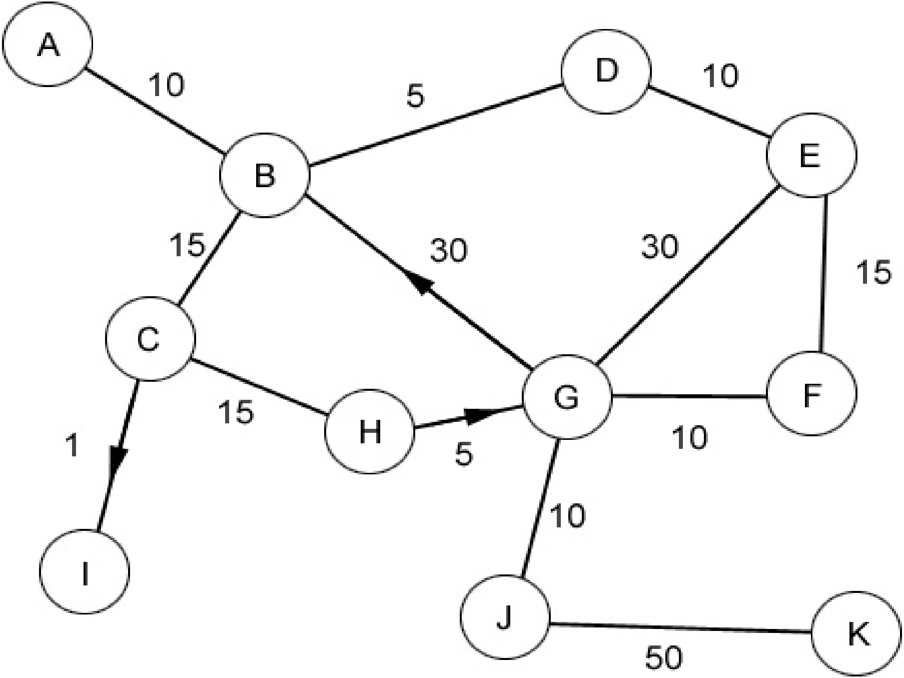
- Sometimes there is a one way road, called a directed edge , and we draw an arrow on it to show which way you can travel along it.
- For instance, if there are two cities A and B, and a line with an arrow from A to B, then we can travel from A to B, but not from B to A.
- Here is an example of a graph, you can't travel from B to A, but you can travel from A to B.
- You can't travel from C to A or from A to C, but you can travel from B to C and from C to B.


B
C
5
30
A

Graph with Cost

- To complicate things even further, we sometimes want to add something called a cost to each edge.
- The idea of a cost is that it indicates how much it would cost to travel down that edge.
- In this graph, most of the people in city A want to get to city C, whereas only a few want to get to city B.
- Unfortunately, both roads are the same size, this means that there are long traffic jams on the road from A to C, it takes about 30 minutes to get there.
- To get from A to B is much easier as most people are going to C, so it only takes 5 minutes.
- The numbers written next to the edges indicate how long


B
C
5
30
A

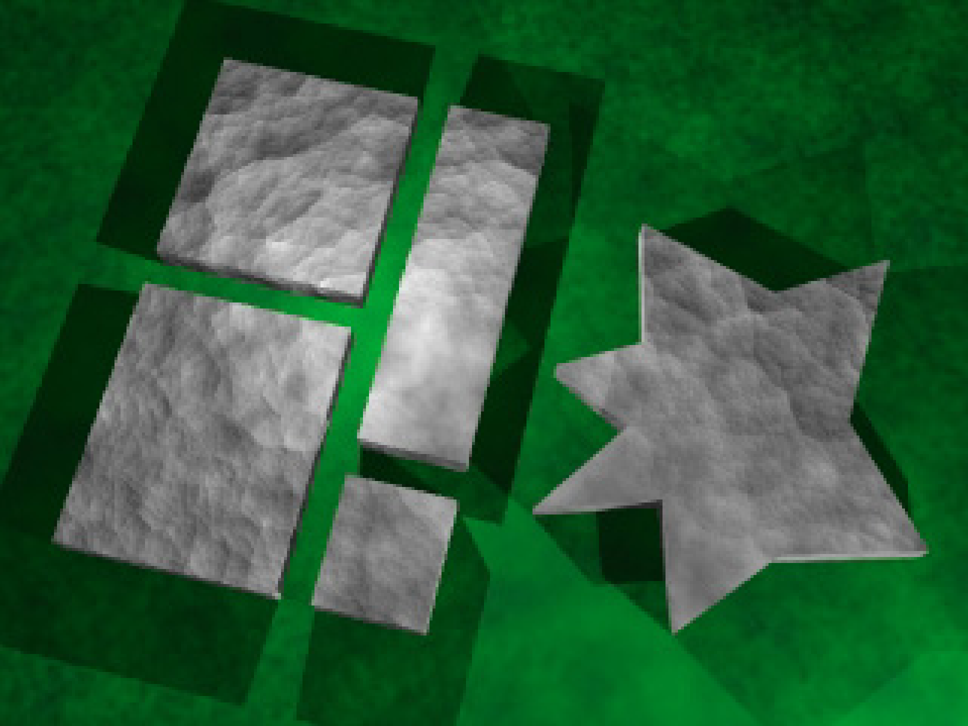

A
B
C
D
E
F
G
H
I
J
K
10
5
10
15
30
30
15
15
1
5
10
10
50

Path Finding I

- Now you know all you need to know to be able to understand path finding.
- How does all this stuff about graphs help the computer guide troops around levels?
- It makes a graph where every interesting point is a node on the graph,
- and every way of walking from one node to another is an edge,
- then it solves the problem
- There are some complications

Path Finding II

- For starters, what are the interesting points?
- You might think that every position on the entire level is interesting
- For most games this would lead to hundreds of thousands of interesting points, and finding the path would take years.
- Instead, the people making the game decide where the interesting points are.
- For instance, if there is a wide open expanse (a big field perhaps), you don't need a node at every point on the field, because the troops can walk in a straight line across the field.



Simulation Games

- The most important thing about simulation games is that they try and make the game like the real world
- Usually this involves physics simulation
- More Details - Later