

Material related to ' Course Defense '

Started: January 9, 2004

Computer Graphics

Description: Scan-conversion algorithms, viewing transformations, visible-surface determination, illumination models, and color theory.

Topics

- | | |
|--------------------------------------------------------------------------------|-----------|
| 1. Introduction to computer graphics hardware and software | (1 week) |
| 2. Graphics primitives: points, lines, polygons, simple curves | (2 weeks) |
| 3. Region fill algorithms | (2 weeks) |
| 4. 2-D viewing: coordinate systems, geometric transformations, clipping | (2 weeks) |
| 5. Complex curves: splines, Bezier curves, fractals | (2 weeks) |
| 6. 3-D viewing: object representations, geometric transformations, projections | (2 weeks) |
| 7. Hidden-line and hidden-surface algorithms | (1 week) |
| 8. Light sources, shading, ray tracing | (1 week) |
| 9. Color models | (1 week) |
| 10. Animation | (1 week) |

Questions:

1. Define the 3-D rotation algorithm
 2. How can we **optimize affine transforms**?
 3. Define scan conversion
 4. Scan conversion of lines
 5. Describe **Line Clipping Algorithms**
 6. Tell me about Scan-conversion algorithms
 7. Tell me about viewing transformations
 8. Discuss the difficulties in "questing" for visual realism
 9. Tell me about visible-surface determination
 10. Tell me about hidden-line and hidden-surface algorithms
 11. Discuss illumination Models
 12. Discuss color theory
 13. Discuss the B-spline
 14. What is a **convex hull**?
 15. What are **affine transformations**?
 16. Discuss the **Window-to-Viewport** transformation (p.211)
 17. What does it mean to talk about **homogeneous coordinates** and matrix representations for 2D transformations?
-

1. Define the 3-D rotation algorithm
2. How can we **optimize or improve the efficiency of affine transforms**?

■ Affine transformations are of the mathematical definition of matrix operations for various types of operations. Some basic affine transformations are scale, translate, rotate and shear.

In order to perform these operations, one would use:

To optimize the matrix operations for performing multiple operations, one can combine operations (such as translation and scaling) together.

Also, optimization can occur for rotations by realizing that a rotation by near 0, can

3. Define scan conversion
4. Scan conversion refers
5. Scan conversion of lines
6. What is the difference between viewport and world-coordinates?
- 7.
8. Describe **Line Clipping Algorithms**
 - a) Brute force method
 - a. We take the object and find the spot where one object intersects another object. We have criteria,
 - i. One point lies inside the clipping rectangle, one does not
 - ii. Both lie inside the clipping rectangle
 1. Trivially Accept the object
 - iii. Both points lie outside the clipping rectangle
 - b. Drawbacks:
 - i. Extremely slow.
 - ii. Line equation for $y=mx+b$ is undefined for vertical lines (slope is undefined)
 - b) Cohen-Sutherland Line-Clipping
 - a. We first perform initial tests to determine if intersection calculations can be avoided. s
 - b.
 - c) Parametric Line Clipping
 - d) Circles, Ellipses
 - e) Polygons

Tell me about Scan-conversion algorithms

Tell me about viewing transformations

Discuss the **difficulties** in “questing” for visual realism

- 1) The complexity of the real world.
 - a. There are surface textures, subtle color gradations, shadows, reflections and slight irregularities in the surrounding objects.
 - b.
- 2)

Tell me about **visible-surface determination**

VSD entails displaying only those parts or surfaces that are visible to the viewer. VSD is also known as *hidden-surface removal*.

Tell me about **hidden-line** and hidden-surface algorithms

Discuss **illumination Models**

Discuss **color theory**

Discuss the **B-spline**

The B-spline consists of curve segments whose polynomial coefficients depend on just a few control points. This is called 'local control'. Thus, moving a control point affects only a small part of a curve. In addition, the time needed to compute the coefficients is greatly reduced. B-splines have the same continuity as natural splines, but do not interpolate their control points.

Is made up of control points, knots, curve segments

Control points = $M+1$

Knots = $M-1$

Curve Segments = $M-2$

What is a **convex hull**?

What does it mean to talk about **homogeneous coordinates** and matrix representations for 2D transformations?

■ **Matrix representations** for translation, scaling, and rotations are

$$P' = T + P,$$

$$P' = S * P,$$

And

$$P' = R * P$$

Respectively.

The problem with these representations is that translation is treated differently than scaling and rotation. Homogeneous coordinates solve this problem by adding an extra parameter to a 2-D vector. So, instead of representing a point with (x,y), we would represent this point with (x,y,W)

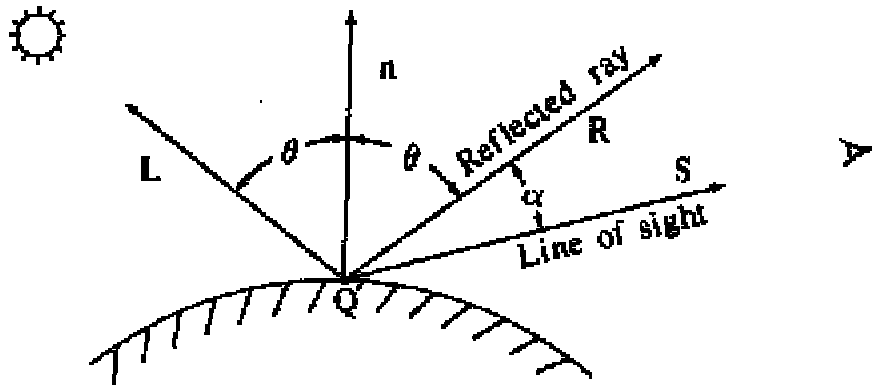
What are **affine transformations**?

Discuss the **Window-to-Viewport** transformation (p.211)

A window-to-viewport transformation involves taking window coordinates (x,y) and translating them to viewport coordinates (u,v).

An issue with window-to-viewport transformations is that if the window and viewport don't have the same dimensions, a non-uniform scaling occurs.

1. General knowledge of computer graphics hardware and software.
2. Graphics primitives: points, lines, polygons, simple curves.
 - a. Line Drawing Algorithms
 - i. DDA
 - ii. $Y=mx+b$ (slope form)
3. Region fill algorithms.
4. 2-D viewing: coordinate systems, geometric transformations, clipping.
5. Complex curves: splines, Bezier curves, and fractals.
 - a. Splines
 - b. Bezier Curves
 - c. Fractals
6. 3-D viewing: object representations, geometric transformations, projections
 - a. Object Representations
 - b. Geometric Transformations
 - c. Projections.
 - i. Perspective
 1. The **center of projection** and **projection plane** are **finite** in distance.
 - ii. Parallel
 1. The **center of projection** and **projection plane** are **infinite** in distance.
 2. Two types: Orthographic and Oblique.
 - a. Orthographic:
 - b. Oblique:
 - i. Two Types: Cavalier and Cabinet
 - ii. Cavalier
 - iii. Cabinet
 - d. Specifying an arbitrary 3D view
 - i. The Projection plane is also called the view plane.
 - ii. Defined by a point on the plane called the View Reference Point (VRP) and a view-plane normal (VPN).
7. Hidden-line and hidden-surface algorithms.
 - a. **Coherence:** Visible surface algorithms can take advantage of coherence – the degree to which parts of an environment or its project exhibit local similarities.
 - i. Object coherence
 - ii. Face coherence
 - iii. Edge coherence
 - iv. Implied Edge Coherence
 - v. Scan-line coherence
 - vi. Area Coherence
 - vii. Depth coherence
 - viii. Frame coherence
 - b. Visible Surface Determination must be done first in 3D space before moving to 2-D space
8. Light sources, shading, ray tracing.
 - a. Diffuse reflection: appears chalky
 - i. Lambertian reflection: another name for diffuse reflection.
 - b. Specular: reflection: is observed on any shiny surface.
 - i. Shining light at an apple highlights a spot, which is specular reflection.
 - ii. Light reflected from the rest of the apple is caused by diffuse reflection.
 - iii. Light reflected specularly from a colorless surface has much of the same color as that of the light source.
 - iv. **Phong Illumination Model**
 1. This is an illumination model for non-perfect reflectors, such as an apple.
 2. It assumes the maximum specular reflectance occurs when $\alpha = 0$ and falls off sharply after that.



3. Reflected intensity be a function of $(\cos \alpha)^n$ with $n \geq 200$ for a shiny surface and n small for a dull surface
4. Angle of reflection = angle of incidence.

c. Shading

- i. **Gouraud:** Also called intensity interpolation shading eliminates intensity discontinuities.
- ii. **Phong:**

9. Color models.

10. Animation.

Image Processing

Description: Image digitization and display, sampling theory, image enhancement and restoration using various spatial, and frequency domain techniques (histogram modification, filtering), Fourier transforms and convolution, image encoding, segmentation, and feature detection.

1. Define the Fourier Transform equation and what it tells us:

It is a measure of the frequency of an image or signal. It is periodic, with a period of $1/T$ where T is the time measure we are currently examining. It is made up of a DC component, which represents the mean of the signal. It is basically the point at which none or very little change in frequency occurs (by the very nature of the definition of mean).

2. Describe Edge Detection:

There are three types of edge detectors:

- (1) 1st Derivative
- (2) 2nd Derivative
- (3) Template Matching

3. What is the mathematical formula for convolution?
4. What are the Morphological Operators?
5. Discuss thresholding (automatic and manual)
6. How would you remove noise from an image?

Statistical Approaches
Wiener filter

Neural Networks

Description: Topics covered include perceptrons, competitive learning, multi-layer networks, back propagation, and selected topics from pattern recognition.

- (1) Describe the fundamental mathematical model of the perceptron or the “perceptron training Rule”:

$$w_i \leftarrow w_i + \Delta w_i$$

Where

$$\Delta w_i = \eta(t - o)x_i$$

Here, t is the target output for the current training example, o is the output generated by the perceptron and η is a positive constant called the learning rate.

2. Describe the Back propagation network:

The back propagation network is a network that is composed of 3 layers, an input, output and hidden layer. The hidden layer is called so because it doesn't have “visible” data to the user, in the same sense as the input and output layers. The make-up of a back prop network consists of weights that are interconnected to the layers.

3. How many **hidden layer nodes** are needed for the **back prop**?

The hidden layer nodes are not well defined, by definition of a Back Propagation neural network. The number of hidden units governs the expressive power of the net - and thus the complexity of the decision

boundary. If the patterns are well-separated or linearly separable, then few hidden units are needed; conversely, if the patterns are drawn from complicated densities that are highly interspersed, then more hidden units are needed.

3. How do we **parallelize** a **Neural Network**?

Using the backprop network, we must consider various techniques/issues:

- What is the function of the Master node and what do the slave nodes do?
- What architecture is the right architecture? (SIMD, MIMD, etc).
- What is the right choice for a memory model? (Shared Memory, PRAM, etc)
- What is the granularity?
- <http://www.mines.edu/students/f/fhood/pllproj/md2000.htm>
-

4. In **Pattern Recognition**, what are the traditional techniques?

- Parametric
- Non-parametric
- Semi-Parametric

Operating Systems

Description: A study of the functions and structures associated with operating systems with respect to process management, memory management, auxiliary storage management, and processor management. Topics include concurrent and distributed computing, deadlock, real and virtual memory, job and processor scheduling, security and protection.

What is **virtual memory**?

Virtual memory is a facility that allows programs to address memory from a logical point of view without regard to the amount of main memory physically available.

Define all the **possible process states**?

Describe how a **DMA system** works:


What function does an **interrupt handler** perform?

Describe how **paging** is done

What is a **process**?

Simplistically, a process is a program that is currently running. A program by itself is not a process.

Describe the **various states of a process**

- (1) **Creation/New:** The process is being created
 - (2) **Running:** Instructions are being executed.
 - (3) **Waiting:** The process is waiting for some event to occur (such as an I/O completion or reception of a signal).
 - (4) **Ready:** The process is waiting to be assigned to a processor.
 - (5) **Terminated:** The process has finished execution.
- 

Advanced Artificial Intelligence

Description: The objective of this course is to provide students with a background in advanced artificial intelligence problem solving methods. Topics covered include: Expert systems, fuzzy logic and fuzzy expert systems, genetic algorithms, case-based reasoning, and current research work on new areas of problem solving.

Advanced Software Engineering

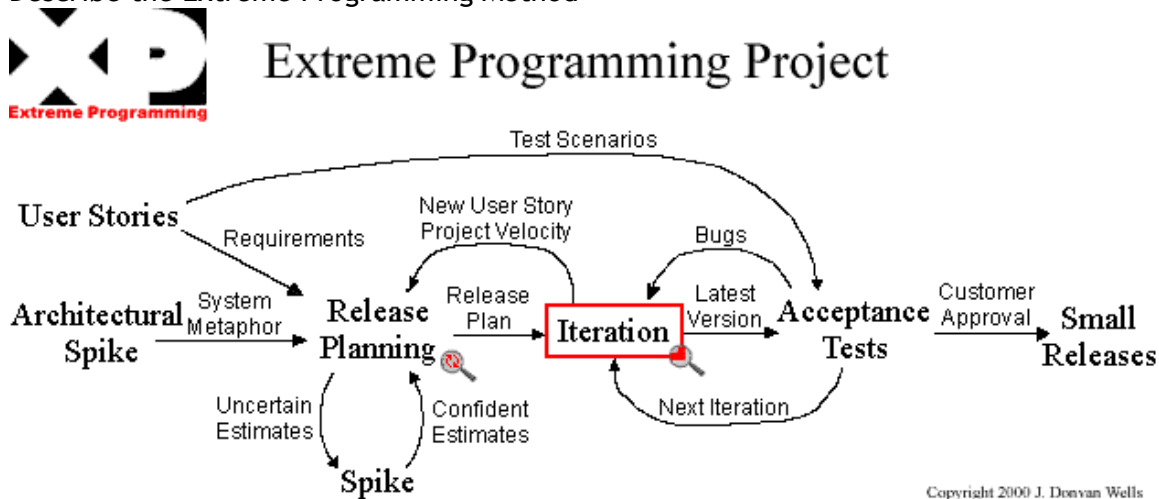
Description: This course covers concepts and techniques within the different phases of the software life cycle: requirements, specifications, design, implementation, testing, operation, and management. The emphasis will be on the study of activities related to software configuration management and maintenance.

1. *Software development process models – development phases, concept of software process and life-cycle, waterfall model, prototyping model, extreme programming model, incremental model, evolutionary model, spiral model, fountain model, and advantages and disadvantages of these models.*
2. *Testing – types of testing including white box and black box methods, design of test cases, reviews, inspections and walkthroughs.*
3. *User Interface design concepts – text, menus/toolbars/icons, messages/error messages/help, and the use of color.*
4. *Object oriented analysis, design and programming – what it is, why use it, the unified modeling language (UML) and the basic tools provided by C++ to accomplish this.*
5. *Project management – PERT/CPM methods*
6. *Metrics – what they are, categories, common metrics*
7. *Software engineering always includes few topics that vary by semester. You should be aware of what they are and be prepared to answer questions. Examples are GUI programming, multimedia design, and web applications.*

Describe and discuss the **COCOMO** method for Software Engineering

Describe the **Waterfall** method of Software Engineering

Describe the Extreme Programming Method



Project Velocity is used to determine either how many stories can be implemented before a given date (time) or how long a set of stories will take to finish (scope).

The base philosophy of release planning is that a project may be quantified by four variables; **scope**, **resources**, **time**, and **quality**. Scope is how much is to be done. Resources are how many people are available. Time is when the project or release will be done. And quality is how good the software will be and how well tested it will be.

Management can only choose 3 of the 4 project variables to dictate, development always gets the remaining variable. Note that lowering quality less than excellent has unforeseen impact on the other 3. In essence there are only 3 variables that you actually want to change. Also let the developer's moderate the customers desire to have the project done immediately by hiring too many people at one time.

User stories serve the same purpose as use cases but are not the same. They are used to create time estimates for the [release planning meeting](#). They are also used instead of a large requirements document. User Stories are written [by the customers](#) as things that the system needs to do for them. They are similar to usage scenarios, except that they are not limited to describing a user interface. They are in the format of about three sentences of text written by the customer in the customer's terminology without techno-syntax.

1. *Software development process models – development phases, concept of software process and life-cycle, waterfall model, prototyping model, extreme programming model, incremental model, evolutionary model, spiral model, fountain model, and advantages and disadvantages of these models.*
 - a. *Development phases*
 - b. *Life-cycle Models*
 - i. *Waterfall*
 - ii. *Prototyping*
 - iii. *Extreme Programming*
 - I. *Risky projects with dynamic requirements are perfect for XP.*
 - iv. *Incremental Model*
 - v. *Evolutionary Model*
 - vi. *Spiral Model*
 - vii. *Fountain Model*
 - c. *Advantages and disadvantages*
2. *Testing – types of testing including white box and black box methods, design of test cases, reviews, inspections and walkthroughs.*
3. *User Interface design concepts – text, menus/toolbars/icons, messages/error messages/help, and the use of color.*
4. *Object oriented analysis, design and programming – what it is, why use it, the unified modeling language (UML) and the basic tools provided by C++ to accomplish this.*
5. *Project management – PERT/CPM methods*
6. *Metrics – what they are, categories, common metrics*
7. *Software engineering always includes few topics that vary by semester. You should be aware of what they are and be prepared to answer questions. Examples are GUI programming, multimedia design, and web applications.*

Neural Networks