

15-462 Computer Graphics I
Lecture 23

Review

End of graphics pipeline
Ray tracing and radiosity
Image processing
Non-photorealistic rendering
Assignment 7 movie

May 2, 2002
Frank Pfenning
Carnegie Mellon University

<http://www.cs.cmu.edu/~fp/courses/graphics/>

Outline

- 12: Physically-Based Modelling
- 13: Texture Mapping
- 14: Clipping and Scan Conversion
- 15: Rasterization
- 16: Ray Tracing
- 17: Spatial Data Structures
- 18: Radiosity
- 19: Global Illumination
- 20: Image Processing
- 21: Scientific Visualization
- 22: Non-Photorealistic Rendering

12 Physically-Based Modelling

- Dynamics
 - Generating motion by applying physical laws
 - Typical: Newton's laws, Hook's law
 - Particles, soft objects, rigid bodies
- Simulates physical phenomena
 - Gravity, momentum, collisions, friction, fluid flow
 - Solidity, flexibility, elasticity, fracture

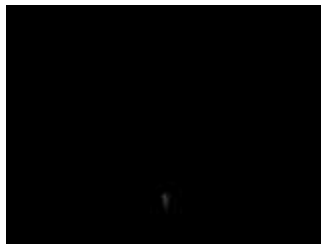
05/02/2002

15-462 Graphics I

3

Particle Systems

- Clouds, smoke, fire, waterfalls
- Each particle rendered as object



Matthew Lewis

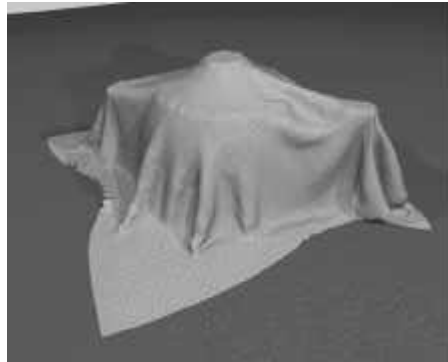
05/02/2002

15-462 Graphics I

4

Spring Forces

- Cloth in 2D, jello in 3D
- Collisions expensive to compute (hierarchical bounding boxes)
- Also: hair
Wooten [Pixar]
- Also: paintbrushes
[Lecture 22]



05/02/2002

15-462 Graphics I

5

Solving Particle Systems

- Use solver for ordinary differential equations
- Discrete approximation (adjust stepsize)
- Euler's method
- Runge-Kutta method
- Specialized method for spring-mass systems
- Constraints
 - Hard: collisions, contact forces, joints
 - Soft: preservation of energy

05/02/2002

15-462 Graphics I

6

13: Texture Mapping

- Adding realism in real time
- Standard applications and bag of tricks
- Texture is 2D image (typically)
- Texture coordinates map image onto surface
- Basic problem: aliasing, perspective
 - Mipmapping: texture depends on resolution
 - Bilinear or trilinear interpolation
- 3D textures (e.g. hair)
- Some tricks:
 - Environment mapping, light maps

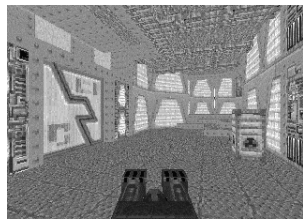
05/02/2002

15-462 Graphics I

7

Light Mapping

- Can paint light map or use radiosity
- Blend several textures

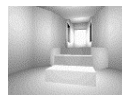


Radiance Texture Map Only



Radiance Texture + Light Map

From Quake



Light Map

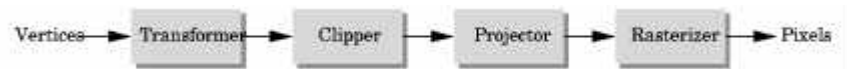
05/02/2002

15-462 Graphics I

8

14: Clipping and Scan Conversion

- Graphics Pipeline, revisited



- Transformation sequence
 1. Camera: From object coordinates to eye coords
 2. Perspective normalization: to clip coordinates
 3. Clipping
 4. Perspective division: to normalized device coords.
 5. Orthographic projection (setting $z_p = 0$)
 6. Viewport transformation: to screen coordinates

05/02/2002

15-462 Graphics I

9

Clipping

- Eliminate objects outside viewing frustum
 - Clipping: in object space
 - Scissoring; in image space
- Cohen-Sutherland clipping: using outcode
- Liang-Barsky clipping: intersection point order
- Polygon clipping
 - Sutherland-Hodgeman clipping pipeline
- Improve efficiency via bounding boxes

05/02/2002

15-462 Graphics I

10

15: Rasterization

- Final step in pipeline (scan conversion)
- Multiple tasks:
 - Filling polygon (inside/outside)
 - Pixel shading (color interpolation)
 - Blending (accumulation, not just writing)
 - Depth values (z-buffer hidden-surface removal)
 - Texture coordinate interpolation (texture mapping)
- Hardware efficiency is critical

05/02/2002

15-462 Graphics I

11

Lines and Polygon

- Incremental algorithm (Bresenham's) for lines
- Fill polygons line by line ("scan conversion")
- Concave polygons
 - Use winding number or even-odd rule
 - Or tessellate into triangles

05/02/2002

15-462 Graphics I

12

Aliasing

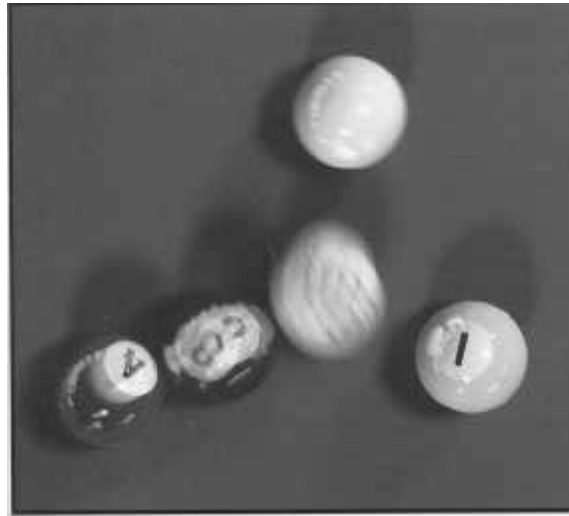
- Artefacts created during scan conversion
- Jagged edges, Moire patterns
- Antialiasing techniques
 - Area averaging [filter]
 - Adaptive supersampling [ray tracing]
 - Jittering
- Temporal aliasing
 - Motion blur through stochastic sampling in time

05/02/2002

15-462 Graphics I

13

Motion Blur Example



T. Porter, Pixar, 1984
16 samples/pixel

05/02/2002

15-462 Graphics I

14

Blending

- Use α channel (RGBA color)
- α determines opacity
- Use for effects such as shadows, blur
- Antialiasing via blending for triangle overlaps

16: Ray Tracing

- Local vs global rendering models
- Object space vs image space
- Three models
 - Graphics pipeline (Phong)
 - Ray tracing
 - Radiosity

Backward Ray Tracing

- From viewer to light
- Basic algorithm
 - Calculate ray/object intersection
 - Cast shadow ray
 - Calculate reflected and transmitted rays
 - Call ray tracer recursively
- Ray-surface intersection for basic shapes
- Support constructive solid geometry (CSG)

05/02/2002

15-462 Graphics I

17

Raytracing Example II



www.povray.org

05/02/2002

15-462 Graphics I

18

17: Spatial Data Structures

- Employed for optimization in various contexts
 - Ray tracing (check fewer ray/object intersections)
 - Radiosity
 - Hidden-surface removal
 - Clipping
 - Collision detection
- Basic bounding volumes
 - Boxes, axis-aligned
 - Boxes, oriented
 - Spheres
 - Finite intersections or unions of above

05/02/2002

15-462 Graphics I

19

Hierarchical Bounding Volumes

- Use tree data structure
- Larger bounding volumes contain smaller ones
- Reduce $O(n)$ to $O(\log(n))$ for certain operations
- May be easy or difficult to compute

05/02/2002

15-462 Graphics I

20

Spatial Subdivision

- For each segment of space, keep list of intersecting surfaces or objects
- Example data structures
 - Regular grids
 - Octrees (axes-aligned, non-uniform)
 - BSP trees (recursive subdivision by planes)
- Efficiency depends on world characteristics
- Example: painter's algorithm using BSP trees

05/02/2002

15-462 Graphics I

21

Constructive Solid Geometry

- Generate complex shapes from simple building blocks
- Particularly applicable for man-made objects
- Efficient with ray tracing
- Use operations
 - Intersection
 - Union (joining objects)
 - Subtraction (e.g., drilling holes, cutting)

05/02/2002

15-462 Graphics I

22

18: Radiosity

- Local illumination: Phong model (OpenGL)
 - Light to surface to viewer
 - No shadows, interreflections
 - Fast enough for interactive graphics
- Global illumination: Ray tracing
 - Multiple specular reflections and transmissions
 - Only one step of diffuse reflection
- Global illumination: Radiosity
 - All diffuse interreflections; shadows
 - Advanced: combine with specular reflection

05/02/2002

15-462 Graphics I

23

Classical Radiosity Method

- Divide surfaces into patches (elements)
- Model light transfer between patches as system of linear equations
- Important assumptions:
 - Reflection and emission are diffuse
 - No participating media (no fog)
 - No transmission (only opaque surfaces)
 - Radiosity is constant across each element
 - Solve for R, G, B separately

05/02/2002

15-462 Graphics I

24

Radiosity Equation

- Assume n surface patches
- Variables
 - A_i Area of element i (computable)
 - B_i Radiosity of element i (unknown)
 - E_i Radiant emitted flux density of element i (given)
 - ρ_i Reflectance of element i (given)
 - F_{ji} Form factor from j to i (computable)

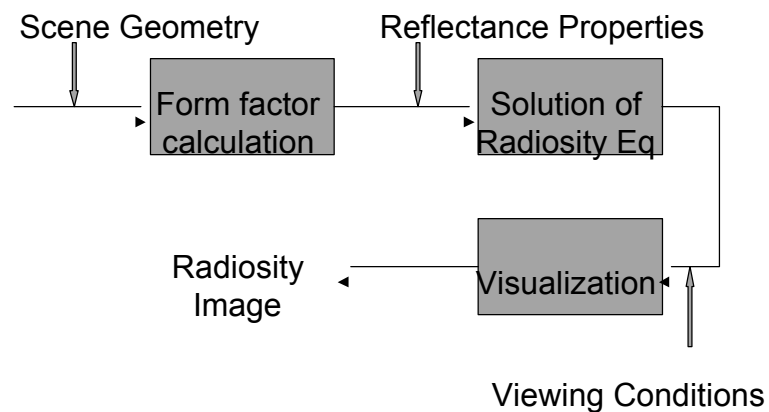
$$A_i B_i = A_i E_i + \rho_i \sum_{j=1}^n F_{ji} A_j B_j$$

05/02/2002

15-462 Graphics I

25

Radiosity “Pipeline”



05/02/2002

15-462 Graphics I

26

Computing Form Factors

- Visibility
 - Hemicube: exploit z-buffer hardware
 - Ray casting
- For inter-visible elements
 - Solve analytically for simple elements
 - Numeric approximation otherwise

05/02/2002

15-462 Graphics I

27

Classical Radiosity Algorithms

- Matrix Radiosity
 - Diagonally dominant matrix
 - Use Gauss-Seidel iterative solution
 - Time and space complexity is $O(n^2)$ for n elements
 - Memory cost excessive
- Progressive Refinement Radiosity
 - Solve equations incrementally with form factors
 - Time complexity is $O(n \cdot s)$ for s iterations
 - Used more commonly (space complexity $O(n)$)

05/02/2002

15-462 Graphics I

28

19: Global Illumination

- Improvements on Radiosity
- Substructuring
 - Subdivide patches into elements, adaptively
 - Analyze transport from patch onto elements
 - Do not consider element-to-element factors
- Progressive Refinement
 - Shoot light instead of gathering light

05/02/2002

15-462 Graphics I

29

Progressive Refinement

- Basic algorithm
 - Initialize emitting element with $B_i = E_i$
 - Initialize others with $B_i = 0$
 - Pick source i (start with brightest)
 - Using hemicube around source, calculate F_{ij}
 - For each $j \neq i$, approximate $B'_j = \rho_j B_i F_{ij} (A_i / A_j)$
 - Pick next source i and iterate until convergence
- Each iteration is $O(n)$
- May or may not keep F_{ij} after each iteration
- Avoid double-counting (“unshot energy”)

05/02/2002

15-462 Graphics I

30

Light Transport and Global Illumination

- Diffuse to diffuse
- Diffuse to specular
- Specular to diffuse
- Specular to specular
- Ray tracing (viewer dependent)
 - Light to diffuse
 - Specular to specular
- Radiosity (viewer independent)
 - Diffuse to diffuse

05/02/2002

15-462 Graphics I

31

Two-Pass Approach

- View-dependent specular is tractable
- View-independent diffuse is tractable
- First pass view independent
 - Enhanced radiosity
- Second pass is view dependent
 - Enhanced ray tracing

05/02/2002

15-462 Graphics I

32

Pass 1: Enhanced Radiosity

- Diffuse transmission (translucent surfaces)
 - Backwards diffuse form factor
- Specular transmission
 - Extended form factor computation
 - Consider occluding translucent surfaces
 - Window form factor
- Specular reflection
 - Create “virtual” (mirror-image) environment
 - Use specular transmission technique
 - Mirror form factor

05/02/2002

15-462 Graphics I

33

Pass 2: Enhanced Ray Tracing

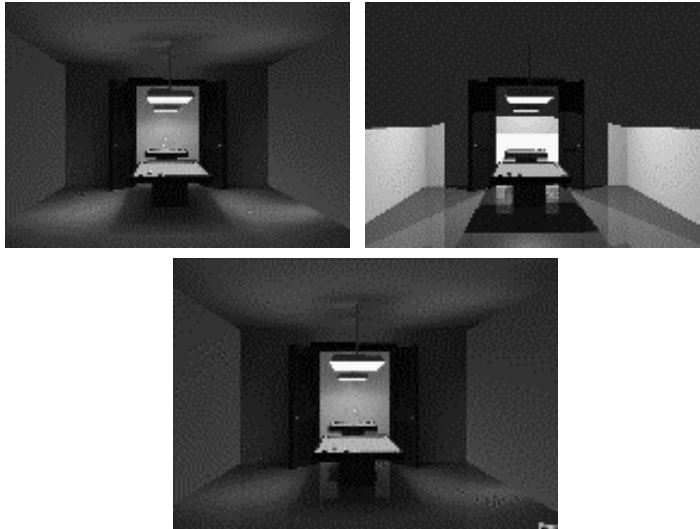
- Classical ray tracing
 - Specular to specular light transport
- For diffuse-to-specular transport:
 - Should integrate incoming light over hemisphere
 - Approximate by using small frustum in direction of ideal reflection
 - Use radiosity of pixels calculated in Pass 1
 - Apply recursively if visible surface is specular

05/02/2002

15-462 Graphics I

34

Two-Pass Radiosity Example



05/02/2002

15-462 Graphics I

35

20: Image Processing

- Display color models
 - 1 bit: black and white display (cf. Smithsonian)
 - 8 bit: 256 colors at any given time via colormap
 - 16 bit: 5, 6, 5 bits (R,G,B), $2^{16} = 65,536$ colors
 - 24 bit: 8, 8, 8 bits (R,G,B), $2^{24} = 16,777,216$ colors
- Image processing
 - Point processing
 - Filtering
 - Compositing
 - Image compression
 - Others [Sullivan guest lecture]

05/02/2002

15-462 Graphics I

36

Linear and Shift-Invariant Filters

- Linear with respect to input signal
- Shift-invariant with respect to parameter

- Convolution in 1D

- a(t) is input signal
 - b(s) is output signal
 - h(u) is filter

$$b(s) = \sum_{t=-\infty}^{+\infty} a(t)h(s-t)$$

- Convolution in 2D

$$b(x, y) = \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} a(u, v)h(x-u, y-v)$$

05/02/2002

15-462 Graphics I

37

Filter Examples

- Blurring filter
- Noise reduction filter
- Edge filter
- Sharpening filter

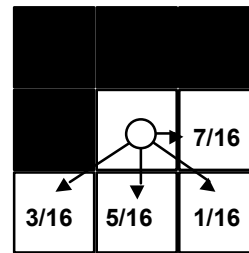
05/02/2002

15-462 Graphics I

38

Dithering

- Compensates for lack of color resolution
- Eye does spatial averaging
- Black/white dithering for gray scale
- Color dithering (calculate RGB separately)
- Floyd-Steinberg error diffusion
 - Scan image in raster order
 - Draw least error value (approximate true color)
 - Divide error into 4 fractions on unwritten pixels



05/02/2002

15-462 Graphics I

39

Image Compression

- Exploit redundancy
 - Coding: some pixel values more common
 - Interpixel: adjacent pixels often similar
 - Psychovisual: some color differences imperceptible
- Distinguish lossy and lossless methods
- Coding redundancy
 - Dictionary to map short codes to long sequences
 - Huffman or Lempel-Ziv-Welch (LZW; gzip)
- Interpixel redundancy
 - Run-length coding, quadtrees, region encoding

05/02/2002

15-462 Graphics I

40

Lossy Compression

- Exploit psychovisual redundancy
- Discrete cosine transform
- JPEG (Joint Photographic Expert Group)
 - Subdivide image into $n \times n$ blocks ($n = 8$)
 - Apply discrete cosine transform for each block
 - Quantize, zig-zag order, run-length code coefficients
 - Use variable length coding (e.g. Huffman)
- Many natural images can be compressed to 4 bits/pixels with little visible error

05/02/2002

15-462 Graphics I

41

21: Visualization

- Generally, no 3D model to start with
- Very large data sets
- Visualize both real-world and simulation data
- Types of data
 - Scalar fields (e.g. x-ray densities)
 - Vector fields (e.g. velocities in wind tunnel)
 - Tensor fields (e.g. stresses in mechanical part)
- Each static or varying through time

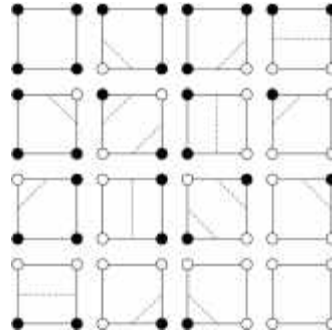
05/02/2002

15-462 Graphics I

42

Marching Squares and Cubes

- Implicit curve $g(x,y) = c$ or surface $f(x,y,z) = c$
- Test g or f on grid points
- Approximate curve or surface based on continuity and smoothness assumptions
- Contour lines (2D)
- Isosurfaces (3D)



05/02/2002

15-462 Graphics I

43

Volume Rendering

- Use voxels (3D “pixels”) and transparency
- Transfer function: data sets to RGBA
 - Psychologically motivated, change interactively
- Volume rendering
 - Volume ray casting (integrate along ray)
 - Splatting (draw shape for each voxel)
 - 3D texture mapping (texture for each layer)

05/02/2002

15-462 Graphics I

44

Visualizing Vector Fields

- Hedgehogs (3D directed line segments)
- Flow lines (color for vector length)
- Animation (partical systems)



05/02/2002

15-462 Graphics I

45

22: Non-Photorealistic Rendering

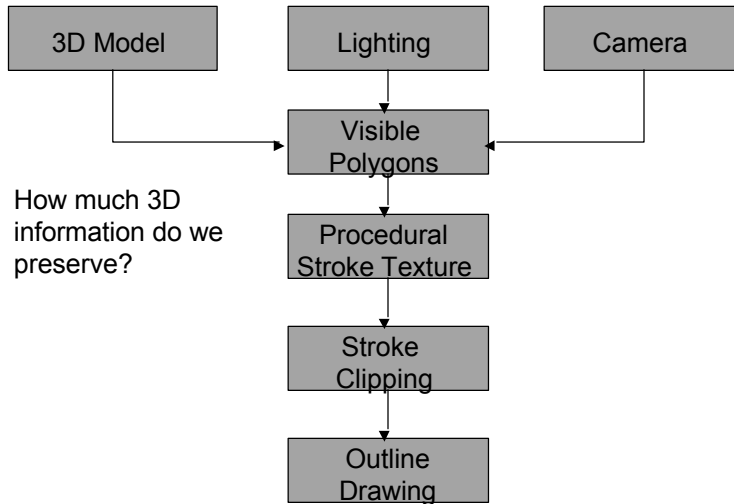
- Cartoons
- Artistic expression in paint, pen-and-ink
- Technical illustrations
- Scientific visualization [lecture 21]

05/02/2002

15-462 Graphics I

46

Pen-and-Ink Illustrations



05/02/2002

15-462 Graphics I

47

Prioritized Stroke Textures

- Technique for limiting human intervention
- Collection of strokes with associated priority
- When rendering
 - First draw highest priority only
 - If too light, draw next highest priority, etc.
 - Stop if proper tone is achieved
- Procedural stroke textures
- Support scaling
- Also applies to non-procedural stroke textures

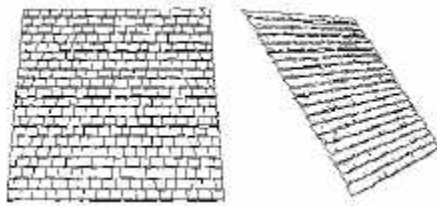
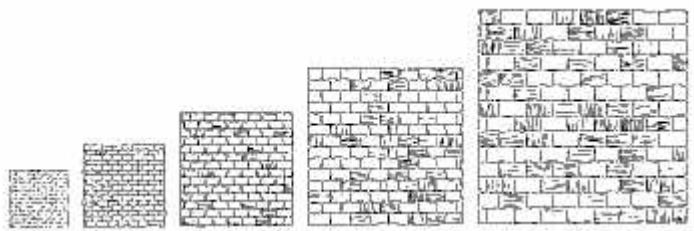
05/02/2002

15-462 Graphics I

48

Stroke Texture Operations

Scaling



Changing Viewing Direction
(Anisotropic)

05/02/2002

15-462 Graphics I

49

Stroke Textures and Indication



Bold strokes
indicate detail
segments

With indication

Without indication



05/02/2002

15-462 Graphics I

50

Painterly Rendering

- Physically simulation
 - Watercolor
 - Oil paint brush strokes
- Painting “over” image
 - Brush stroke parameters determine style
 - Superficially adequate, lack of artistic sensitivity

05/02/2002

15-462 Graphics I

51

Layered Painting

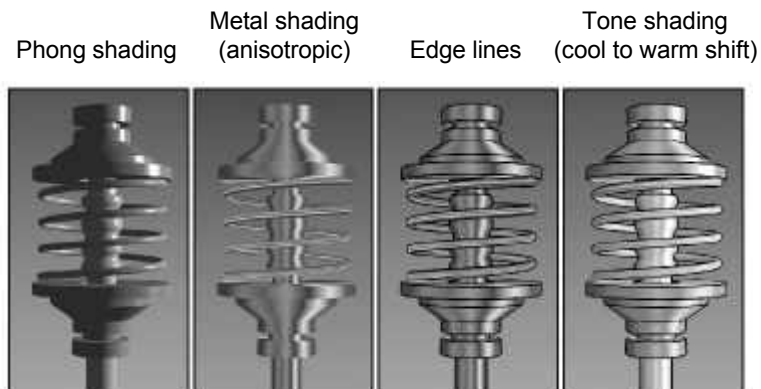


05/02/2002

15-462 Graphics I

52

Technical Illustration Example



05/02/2002

15-462 Graphics I

53

Summary


- 12: Physically-Based Modelling
- 13: Texture Mapping
- 14: Clipping and Scan Conversion
- 15: Rasterization
- 16: Ray Tracing
- 17: Spatial Data Structures
- 18: Radiosity
- 19: Global Illumination
- 20: Image Processing
- 21: Scientific Visualization
- 22: Non-Photorealistic Rendering

05/02/2002

15-462 Graphics I

54

Assignment 7 Movie



Michael Maxim

05/02/2002

15-462 Graphics I

55

Announcements

- Assignment 8
 - Was due before lecture
 - Model solutions available tomorrow (Friday) morning
- Final
 - Monday, May 6, 1:00-4:00, WeH 7500
 - Emphasis on 2nd half of course (this lecture)
 - Problems similar to midterm and assignments
 - Open book, open notes, no laptop
 - Worth 250 points
- Please fill out TA evaluation forms

05/02/2002

15-462 Graphics I

56

Finally...

- It's been fun --- thanks!