Final Exam Review Session

CS148: Introduction to Computer Graphics and Imaging
Stanford University
Info about the exam

- Wednesday, Dec 14th, 7-10pm
- Braun Auditorium (Mudd Chemistry Building)
- Open notes, book, computer (no network)
## Last office hours

<table>
<thead>
<tr>
<th>Day</th>
<th>Time</th>
<th>Location</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Friday</td>
<td>2:15 - 4:15</td>
<td>Gates B26B</td>
<td>Matt</td>
</tr>
<tr>
<td>Monday</td>
<td>10:50 - 12:50</td>
<td>Gates B24B</td>
<td>Daniel</td>
</tr>
<tr>
<td></td>
<td>1 - 3</td>
<td>Gates 398</td>
<td>Crystal</td>
</tr>
<tr>
<td></td>
<td>4 - 6</td>
<td>Gates B26</td>
<td>Jorge</td>
</tr>
<tr>
<td>Tuesday</td>
<td>9 - 11</td>
<td>Gates 381</td>
<td>Katherine</td>
</tr>
<tr>
<td></td>
<td>12 - 2</td>
<td>Gates B24B</td>
<td>Daniel</td>
</tr>
<tr>
<td></td>
<td>4 - 6</td>
<td>Bytes Cafe</td>
<td>Alexis</td>
</tr>
<tr>
<td>Wednesday</td>
<td>10 - 12</td>
<td>Gates B26</td>
<td>Matt</td>
</tr>
<tr>
<td></td>
<td>1 - 3</td>
<td>Gates 398</td>
<td>Crystal</td>
</tr>
</tbody>
</table>
Topics

- Light and Color
- Cameras
- Displays
- Sampling
- Image Compression
- Programmable GPUs
- Rendering
- Ray Tracing
- Modeling
Light and Color

• Physics
• Perception
• Color spaces
Physics of Color

- Linear combination of light of different frequencies makes up most visible light. Objects are colored differently based on which frequencies they emit/reflect. Two fundamental operations: adding and multiplying spectra.

- Emissive/additive objects (eg, CRTs) add component spectra to make a final output color:
  
  \[ L(\lambda) = S_1(\lambda) + S_2(\lambda) + \ldots \]

- Reflective/subtractive objects (pigments) remove component colors by multiplying spectra.
  
  \[ L(\lambda) = \text{Source}(\lambda) \times \text{Pigment}(\lambda) \]

- Measurement is an integration of the product of the spectrum times a sensor response function.

  \[ M = \int \text{Source}(\lambda) \times \text{Response}(\lambda) d\lambda \]
Human perception of color

- Most humans have three types of color receptors (cones) and one type of luminance receptor (rods). Cones perceive different colors because of different response functions.

- **Trichromatism** - any color representable as linear combination of only three colors

- **Metamerism** - many different spectra resolve the same perceived color because we have only three types of cone

- **Color blindness** - missing cone types prevent perception of certain colors as different (super-metamerism!)
Color spaces

- Perceptual organization (BW, RG, YB)
- Hue Saturation Value (RGB to HSV)
Cameras

- Pinhole camera
- Field of view (FOV)
- Perspective projection (homogenous coords, perspective matrix, frustum)
- Z-buffer
- Focus, depth of field (DOF)
- Exposure
- Response curve
- Aperture/shutter
Pinhole Camera

- Zero aperture
- Infinite depth of field
- Field of view determined by focal length and size of film

\[
\tan \frac{\text{fov}}{2} = \frac{\text{filmdiag}/2}{f}
\]
Perspective projection

- Non linear: homogeneous coordinates (4x4 matrix)
- View frustum: mapped to a cube by frustum matrix
- Keeps depth for z-buffering

\[
x' = -f \frac{x}{z}
\]
\[
y' = -f \frac{y}{z}
\]
Real camera, lenses

- Depth of field, focus, circle of confusion

- Exposure proportional to aperture^2
- Aperture vs shutter speed (for same exposure)
Displays

- Spatial resolution
- Temporal resolution
- HDR, Tone mapping
Human perception

- **Spatial resolution**: due to photoreceptor density in retina. Rods (100 million), Cones (5 million).

- **Contrast sensitivity**

- **Temporal resolution**
  Flicker fusion around 20-30 Hz in low ambient light, 80 Hz in bright ambient light.
Tone Mapping

- Sensation vs Intensity (nonlinear) $S = I^{0.33}$
- Contrast: max/min
- World: $100,000,000,000:1$
- People: $100:1$
- Displays non linear $I \sim (V - V_b)^Y$
- Nonlinearities in sensation and display cancel
- Display range: $80:1 - 1000:1$
- Linear mapping
- Logarithmic mapping
Sampling

- Frequency domain vs spatial domain
- Filters, convolution
- Sampling
- Nyquist frequency
- Aliasing, Antialiasing
Fourier Transform, Filtering

- Fourier transform converts spatial domain to frequency domain
  - Fast variation / hard edges: high frequencies
  - Slow variation / constants: low frequencies

- Linear filters are multiplications in frequency domain (lowpass filter) or convolutions in spatial domain

- Filter kernels localized in space are spread out in frequency domain and vice-versa

- Filtering complexity on NxN image:
  - Spatial domain, K-element kernel: $O(KN^2)$
  - Frequency domain, L-frequency kernel: $O(L + N^2 \lg N)$
Nyquist frequency and sampling

• Sampling: evaluating a continuous function (e.g., a triangle in space) at a point or discrete set of points (e.g., fragment centers)

• Sampling theorem: if a continuous function contains no frequencies higher than $F$, then it is completely determined by its samples taken at a rate of $2F$

• Nyquist sampling criterion: Must sample a function at twice its highest frequency to recover it, or else we will introduce artifacts. Frequencies in the input signal at higher than half the sampling frequency will appear as aliases onto lower frequencies in the sampled output.
Aliasing, Antialiasing

• Aliasing

• Antialiasing: Lowpass filter the input before sampling it to remove high frequencies. Or sample more.
Entropy, Huffman Coding

• Entropy: average number of bits required for optimal code from a given probability distribution
  \[- \sum_{i=1}^{n} p(x_i) \log_2 p(x_i)\]

• Huffman coding: less frequent values have longer codes than more frequent values, recursive tree construction
  - each symbol is a node
  - merge two lowest probability nodes
  - repeat until only one node exists
  - code given by path from root
Lossy Compression

- Distribute error to higher frequencies (low frequencies more common, visual system less sensitive to high freq)
- Quantization: use less bits to store information
- Error quantified by \( \text{PSNR} = 10 \log_{10} \left( \frac{I_{\text{max}}^2}{\text{MSE}} \right) \)
- Change of basis: concentrate information in a useful way to throw away low magnitude components easily

Discrete cosine (JPEG): frequency domain

Wavelet basis (JPEG2000): multiscale decomposition
GPUs

- OpenGL Pipeline: stages
- Programmable stages (vertex and fragment shaders)
- GLSL
# The (old) OpenGL Pipeline

<table>
<thead>
<tr>
<th>Stage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>maintain graphics state, aggregate per vertex data</td>
</tr>
<tr>
<td>Vertex</td>
<td>vertex and normal transform, tex coord generation</td>
</tr>
<tr>
<td>Assembly</td>
<td>combine vertices into primitives, clip, perspective proj, culling</td>
</tr>
<tr>
<td>Rasterization</td>
<td>sample triangles into fragments, interpolate colors, normals, etc.</td>
</tr>
<tr>
<td>Fragment</td>
<td>use normal, color, texture to assign color to fragments</td>
</tr>
<tr>
<td>Framebuffer ops</td>
<td>test alpha, depth, compositing &gt; final framebuffer</td>
</tr>
<tr>
<td>Display</td>
<td>gamma correction, send framebuffer back to the CPU</td>
</tr>
</tbody>
</table>
Programmable stages

• Vertex shaders: per-vertex operations
  - modelview transform
  - displacement (GPU animation)

• Fragment shaders: per-fragment operations
  - shading
  - image processing

• Limitations: memory
  - no access to neighbors
  - stack space
  - framebuffer read/write
Rendering

• Reflection models:
  Phong shading

• Normals:
  compute, transform

• Environment map

• Displacement, normal map
Reflection models

- Ambient, diffuse, specular lighting
- Mirror: reflected view vector
- Specular: Phong \((R.V)^s\) or Blinn-Phong \((N.H)^s\)
- Diffuse: Lambert’s Law: \(L.N\)
Hacks to render cool stuff

- Environment mapping (light probe)
- Normal mapping
- Displacement mapping
Ray Tracing

- Shoot a ray from the eye to the scene
- Find closest intersection
- Shoot a shadow ray
- Reflect the ray if needed
- Shade the point: ambient + diffuse + specular + reflection
- Transformations
- Light types: point, directional
Modeling

• Represent a 3D shape: mesh, bicubic splines, subdivision surfaces
• Create 3D shapes: CAD, scanner, procedural
• Manipulate: deform/skin/animate, smooth, compress, set unions

• Store in data structures: polygon, points, topological
• Subdivision surfaces: Loop, Catmull-Clark
Geometry Storage

- Polygon list: simple but redundant
- Points+Polygons: reduces memory usage
- Topological data structures:
  - access to neighbors (normal calculation, subdivision)
  - edit the geometry (add new vertices)
- (Winged-Edge Representation)
Subdivision Surfaces

- 2D generalization of Bezier subdivision: refine polygon meshes by adding vertices
- Triangles meshes: Loop subdivision
- (Quad meshes: Catmull-Clark subdivision)
What you don’t need to know
(for the exam)

• Cameras
  - Light field camera
  - Sensors
• Displays
  - Display technologies
• Sampling
  - Extra slides on supersampling
• GPUs
  - GPU Computing
• Rendering
  - Translucent materials
• Modeling
  - Winged-Edge representation
  - Catmull-Clark subdivision
• Animation
That's it?
No...

- Lecture slides
- Assignments
- Readings
- Past exams
Good Luck!

- Don’t work too much
- Enjoy your break
- See you next quarter