Volume Rendering, Part 2

Hank Childs, University of Oregon
Announcements

• VisIt lecture: will be posted (sorry)
• Grading: will happen soon
• Projects:
  – ~30 hours
  – Aim for something worthy of your resume
• Pre-defined project: posted tonight
• Volume rendering lectures: today & Weds
• Weds: yes, there’s class
Review
X-rays

Emitter

Film/image

[Image of X-ray of a hand]
Volume rendering

• Important visualization technique for 3D data
• Use combination of color and transparency to see entire 3D data set at one time.

There are multiple ways to do volume rendering. I will describe one way today (raycasting). That will help explain the technique. I will describe alternate ways on Friday.
Volume rendering overview

Camera

Pixels on the screen

3D data
Volume rendering overview

Ray casting game plan:
For every pixel on the screen,
Find ray for that pixel
Intersect volume with ray
Calculate color from intersection
Assign color to pixel
Outline

• Find Ray For That Pixel
• Intersect Volume With Ray
• Calculate Color From Intersection
• Assign Color To Pixel
Outline

• Find Ray For That Pixel
• Intersect Volume With Ray
• Calculate Color From Intersection
• Assign Color To Pixel
How do we specify a camera?

The “viewing pyramid” or “view frustum”.

Frustum: In geometry, a frustum (plural: frusta or frustums) is the portion of a solid (normally a cone or pyramid) that lies between two parallel planes cutting it.

class Camera
{
    public:
        double near, far;
        double angle;
        double position[3];
        double focus[3];
        double up[3];
};
From Pixels to Rays

\[ \mathbf{r} = \frac{\text{look} \times \mathbf{u}}{|\text{look} \times \mathbf{u}|} \]

\[ \mathbf{r} = \frac{\text{look} \times \mathbf{v}}{|\text{look} \times \mathbf{v}|} \]

\[ \Delta x = \frac{2 \tan(\text{fov}_x / 2) \mathbf{r}}{W} \]

\[ \Delta y = \frac{2 \tan(\text{fov}_y / 2) \mathbf{r}}{H} \]

\[ d(i, j) = \frac{\text{look}}{|\text{look}|} + \frac{(2i + 1 - W) \mathbf{r}}{2 \Delta x} + \frac{(2j + 1 - H) \mathbf{r}}{2 \Delta y} \]

This answers the “find ray for this pixel” question.
Outline

• Find Ray For That Pixel
• Intersect Volume With Ray
• Calculate Color From Intersection
• Assign Color To Pixel
What is the result of the ray-volume intersection?
Ray-Volume Intersection

How would you calculate this intersection?
Face Intersections

Can we calculate the intersections at the faces?
Can we calculate the data at the faces?

Is this a good idea?
Why or why not?

(it is not a good idea if we interpolate between the face values)
How do we fix this?
Ray-Volume Intersection: sampling

Sampling is the most common method for “ray-casting” volume rendering

Do we know how to do this sampling?
How to sample quickly

• Multiple strategies.

• For now, similar as before:
  – Find first cell intersected
    • Intersection is at a face
  – Find where ray exits that cell
    • Are there samples within the cell? Then sample them?
  – Go to next cell (which shares a face) and repeat
  – Keep going until you exit the volume, one cell at a time, and see what samples it covers

Approximately how many samples will we calculate?
Outline

• Find Ray For That Pixel
• Intersect Volume With Ray
• Calculate Color From Intersection
• Assign Color To Pixel
Refresher: Color Maps

Pseudocolor
Var: hardyglobal
Units: Joules
Transfer function

• Assigns a color and an opacity to each scalar value.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Abstraction for assigning color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pseudocoloring</td>
<td>Color Map</td>
</tr>
<tr>
<td>Volume Rendering</td>
<td>Transfer Function</td>
</tr>
</tbody>
</table>
Transfer Function

Volume
Var: hardyglobal
Units: Joules

Max: 5.890
Min: 1.096
“Alpha Channel”

• Represents opacity
  – 1.0 or 255: fully opaque
  – 0: fully transparent

• Stored alongside RGB
  – Referred to as RGBA

• Floating point (1.0) vs byte (255):
  – Precision vs uniformity with RGB & performance
Applying a transfer function

<table>
<thead>
<tr>
<th>Sample</th>
<th>Scalar Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5.8</td>
</tr>
<tr>
<td>1</td>
<td>4.7</td>
</tr>
<tr>
<td>2</td>
<td>5.8</td>
</tr>
<tr>
<td>3</td>
<td>3.5</td>
</tr>
<tr>
<td>4</td>
<td>1.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R</th>
<th>G</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>255</td>
<td>0</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>255</td>
<td>255</td>
<td>0</td>
<td>0.75</td>
</tr>
<tr>
<td>255</td>
<td>0</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>0</td>
<td>255</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>255</td>
<td>0</td>
</tr>
</tbody>
</table>

Quiz: calculate the results from transfer function for each sample
Transparency

• Quiz: If you have a red square that is 50% opaque in front of a black background, what color would you see?
  – Represent your answer in terms of (R, G, B)

Answer: (128, 0, 0)
Transparency

• If you have an opaque red square in front of a blue square, what color would you see?
  – Red

• If you have a 50% transparent red square in front of a blue square, what color would you see?
  – Purple

• If you have a 100% transparent red square in front of a blue square, what color would you see?
  – Blue
Formula For Transparency

- **Front** = (Fr, Fg, Fb, Fa)
  - a = alpha, transparency factor
    - Sometimes percent
    - Typically 0-255, with 255 = 100%, 0 = 0%
- **Back** = (Br, Bg, Bb, Ba)
- **Equation** = (Or = Fa*Fr+(1-Fa)*Ba*Br,
  Og = Fa*Fg+(1-Fa)*Ba*Bg,
  Ob = Fa*Fb+(1-Fa)*Ba*Bb,
  Oa = Fa+(1-Fa)*Ba)

Alpha component is important! Any observations?
Representing Colors

• Two ways of representing colors
  – Pre-multiplication
  – Straight RGBA (non-premultiplied)

• Slides use straight RGBA.
  – Will describe pre-multiplication afterwards

Wikipedia page is great reference:
http://en.wikipedia.org/wiki/Alpha_compositing
Transparency

• If you have an 25% transparent red square (255,0,0) in front of a blue square (0,0,255), what color would you see (in RGB)?
  – (192,0,64)

• If you have an 25% transparent blue square (0,0,255) in front of a red square (255,0,0), what color would you see (in RGB)?
  – (64,0,192)
Calculating Color

<table>
<thead>
<tr>
<th>Sample</th>
<th>Scalar Value</th>
<th>R</th>
<th>G</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5.8</td>
<td>255</td>
<td>0</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>1</td>
<td>4.7</td>
<td>255</td>
<td>255</td>
<td>0</td>
<td>0.75</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1.1</td>
<td>0</td>
<td>0</td>
<td>255</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: we will talk more about alpha

- Equation = \((Fa \times Fr + (1-Fa) \times Ba \times Br, \newline Fa \times Fg + (1-Fa) \times Ba \times Bg, \newline Fa \times Fb + (1-Fa) \times Ba \times Bb, \newline Fa + (1-Fa) \times Ba)\)

Apply this equation over and over until you run out of samples (then use background color)
Game Plan

- Initialize pixel color / alpha with `TransferFunction(first sample)`
- Iterate over remaining samples
  - Front-to-back order
  - pixel color / alpha updated each time using composite function
Outline

• Find Ray For That Pixel
• Intersect Volume With Ray
• Calculate Color From Intersection
• Assign Color To Pixel
Assign Color To Pixel

• Allocate a buffer for storing RGB values
  – Buffer should have one RGB for every pixel on the screen.

• As you calculate color for a ray, assign that color to its corresponding buffer entry

• When you have all of the colors, put that image up on the screen, as if you had rendered it using graphics cards.
Volume rendering overview

Ray casting game plan:
For every pixel on the screen,
Find ray for that pixel
Intersect volume with ray
Calculate color from intersection
Assign color to pixel

Which of these 4 steps were easy? Which were hard?
Ray-Volume Intersection: sampling

Camera

Pixels on the screen

What happens when we change the sampling rate?

Imagine if we had half or twice as many transparent squares...

Should the picture change if we change the sampling rate?
Opacity Adjustment

• Consider ray with this profile

What is the opacity of these three samples together?
Opacity Adjustment

• Consider ray with this profile

New interpretation: opacity is for a range, not a point. If you double the sampling, then the range is reduced. Typically think of opacity for a given sampling rate and then adjust as sampling rate changes.

What is the opacity of these five samples together?
Opacity Correction

The assigned opacity also depends on the sampling rate. For example, when using fewer slices, the opacity has to be scaled up, so that the overall intensity of the image remains the same. Equation 3 is used for correcting the transfer function opacity whenever the user changes the sampling rate \( s \) from the reference sampling rate \( s_0 \):

**Equation 3 Formula for Opacity Correction**

\[
A = 1 - (1 - A_0)^{s_0/s}
\]