Volume rendering

UCSD CSE 167
Tzu-Mao Li
Course evaluation!

- Please let us know if you have any feedback regarding the course

- https://academicaffairs.ucsd.edu/Modules/Evals?e11081128
Today: foggy and transparent stuff

https://en.wikipedia.org/wiki/Sunbeam
"... in 10 years, all rendering will be volume rendering."
Jim Kajiya at SIGGRAPH '91

A Survey of Algorithms for Volume Visualization
T. Todd Elvins
San Diego Supercomputer Center
Surface vs volume rendering

clear boundary

fuzzy boundary
Volumes can be useful for representing complex materials


https://stephenlombardi.github.io/projects/neuralvolumes/
Physics of volume rendering: how light interact with particles

more particles -> lights are blocked more, less particles -> lights are blocked less
In practice, instead of storing individual particles, we store the density $\sigma(x)$.

Here is a table showing different values of $\sigma$:

<table>
<thead>
<tr>
<th>$\sigma = 0$</th>
<th>$\sigma = 1$</th>
<th>$\sigma = 1$</th>
<th>$\sigma = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 0$</td>
</tr>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 0$</td>
</tr>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 1$</td>
<td>$\sigma = 1$</td>
<td>$\sigma = 0$</td>
</tr>
</tbody>
</table>
In addition, we store a color $C$ per voxel

<table>
<thead>
<tr>
<th>$\sigma = 0$</th>
<th>$\sigma = 1$</th>
<th>$\sigma = 1$</th>
<th>$\sigma = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C = (0,0,0)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0,0,0)$</td>
</tr>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 0$</td>
</tr>
<tr>
<td>$C = (0,0,0)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0,0,0)$</td>
</tr>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 0$</td>
</tr>
<tr>
<td>$C = (0,0,0)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0,0,0)$</td>
</tr>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 1$</td>
<td>$\sigma = 1$</td>
<td>$\sigma = 0$</td>
</tr>
<tr>
<td>$C = (0,0,0)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0.5,0.5,1)$</td>
<td>$C = (0,0,0)$</td>
</tr>
</tbody>
</table>
Goal: render a volume given a camera pose

we need to know how to combine all the colors given a ray!

<table>
<thead>
<tr>
<th>( \sigma = 0 )</th>
<th>( \sigma = 1 )</th>
<th>( \sigma = 1 )</th>
<th>( \sigma = 0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C = (0,0,0) )</td>
<td>( C = (0.5,0.5,1) )</td>
<td>( C = (0.5,0.5,1) )</td>
<td>( C = (0,0,0) )</td>
</tr>
<tr>
<td>( \sigma = 0 )</td>
<td>( \sigma = 10 )</td>
<td>( \sigma = 10 )</td>
<td>( \sigma = 0 )</td>
</tr>
<tr>
<td>( C = (0,0,0) )</td>
<td>( C = (0.5,0.5,1) )</td>
<td>( C = (0.5,0.5,1) )</td>
<td>( C = (0,0,0) )</td>
</tr>
<tr>
<td>( \sigma = 0 )</td>
<td>( \sigma = 1 )</td>
<td>( \sigma = 1 )</td>
<td>( \sigma = 0 )</td>
</tr>
<tr>
<td>( C = (0,0,0) )</td>
<td>( C = (0.5,0.5,1) )</td>
<td>( C = (0.5,0.5,1) )</td>
<td>( C = (0,0,0) )</td>
</tr>
</tbody>
</table>
Physics of volume rendering

let's parameterize the energy carried by a light with $L(t)$
Physics of volume rendering

let's parameterize the energy carried by a light with $L(t)$

every time a light hits a particle, a fraction of its energy get absorbed
Physics of volume rendering

let’s parameterize the energy carried by a light with $L(t)$

every time a light hits a particle, a fraction of its energy get absorbed.

we model this using an ODE

$$\frac{d}{dt}L(t) = - \sigma(t)L(t)$$
Physics of volume rendering

let’s parameterize the energy carried by a light with $L(t)$

every time a light hits a particle, a fraction of its energy get absorbed

we model this using an ODE

$$\frac{d}{dt}L(t) = -\sigma(t)L(t) + C(t)$$

adding color $C$ if the particles emit lights
Physics of volume rendering

\[
\frac{d}{dt} L(t) = -\sigma(t)L(t) + C(t)
\]

absorption

emission

want to “solve” this ODE
Physics of volume rendering

\[
\frac{d}{dt} L(t) = - \sigma(t)L(t) + C(t)
\]

(absorption)

\[
\frac{d}{dt} L(t) = - aL(t)
\]

(emission)

want to “solve” this ODE

a simpler ODE has the following analytical solution

\[
L(t) = \exp(-at)L(0)
\]
Physics of volume rendering

\[ \frac{d}{dt}L(t) = -\sigma(t)L(t) + C(t) \]

absorption

emission

\[ L(t) = \int_{0}^{t} T(t')C(t')dt' \]

\[ T(t) = \exp \left( -\int_{0}^{t} \sigma(t')dt' \right) \]
In practice, we often turn integrals into discrete sums

\[ L(t) = \int_0^t T(t')C(t')dt' \]

\[ T(t) = \exp\left(-\int_0^t \sigma(t')dt'\right) \]

\[ L = \sum_i T_i C(t_i) \Delta_t \]

\[ T_i = \exp\left(-\sum_{j=0}^i \sigma(t_j) \Delta_t\right) \]
Intuition of the volume rendering equation

\[ L = \sum_i T_i C(t_i) \Delta_t \]

\[ T_i = \exp \left( - \sum_{j=0}^{i} \sigma(t_j) \Delta_t \right) \]
Intuition of the volume rendering equation

\[ C = \text{light}, \ T = \text{amount of light passes through} \]

\[ L = \sum_i T_i C(t_i) \Delta t \]

\[ T_i = \exp \left( - \sum_{j=0}^{i} \sigma(t_j) \Delta t \right) \]
Intuition of the volume rendering equation

C = light, T = amount of light passes through

\[ L = \sum_{i} T_{i} C(t_{i}) \Delta t \]

\[ T_{i} = \exp \left( - \sum_{j=0}^{i} \sigma(t_{j}) \Delta t \right) \]
Intuition of the volume rendering equation

\[ C = \text{light}, \ T = \text{amount of light passes through} \]

\[ L = \sum_i T_i C(t_i) \Delta t \]

\[ T_i = \exp \left( - \sum_{j=0}^{i} \sigma(t_j) \Delta t \right) \]
The volume rendering algorithm (aka raymarching)

for each pixel, shoot a ray from camera origin

\[
\begin{array}{cccc}
\sigma = 0 & \sigma = 1 & \sigma = 1 & \sigma = 0 \\
C = (0,0,0) & C = (0.5,0.5,1) & C = (0.5,0.5,1) & C = (0,0,0) \\
\sigma = 0 & \sigma = 10 & \sigma = 10 & \sigma = 0 \\
C = (0,0,0) & C = (0.5,0.5,1) & C = (0.5,0.5,1) & C = (0,0,0) \\
\sigma = 0 & \sigma = 10 & \sigma = 10 & \sigma = 0 \\
C = (0,0,0) & C = (0.5,0.5,1) & C = (0.5,0.5,1) & C = (0,0,0) \\
\sigma = 0 & \sigma = 1 & \sigma = 1 & \sigma = 0 \\
C = (0,0,0) & C = (0.5,0.5,1) & C = (0.5,0.5,1) & C = (0,0,0) \\
\end{array}
\]

\[
L_j = \sum_{i=0}^{j} T_i C(t_i) \Delta t
\]

\[
T_i = \exp \left( -\sum_{0}^{i} \sigma(t_i) \Delta t \right)
\]
The volume rendering algorithm (aka raymarching)

for each pixel, shoot a ray from camera origin

sample N steps along the ray

\[
\begin{align*}
L_j &= \sum_{i=0}^{j} T_i C(t_i) \Delta t \\
T_i &= \exp\left(-\sum_{0}^{i} \sigma(t_i) \Delta t\right)
\end{align*}
\]
The volume rendering algorithm
(aka raymarching)

```
sigma_sum = 0
color = 0
delta_t = ...
for i in range(N):
    t = ... # distance to ray origin
    C = voxel_C(ray(t))
    sigma = voxel_Sigma(ray(t))
    sigma_sum += sigma
    T = exp(-sigma_sum * delta_t)
    color += T * C * delta_t
```

\[ L_j = \sum_{i} T_i C(t_i) \Delta_t \]
\[ T_i = \exp\left(-\sum_{0}^{i} \sigma(t_i)\Delta_t\right) \]
Equivalence to alpha blending

\[ L = \sum_i T_i C(t_i) \Delta t \]

\[ T_i = \exp \left( - \sum_{j=0}^{i} \sigma(t_j) \Delta t \right) \]
Equivalence to alpha blending

\[ L = \sum_{i} T_i C(t_i) \Delta t \]
\[ T_i = \exp \left( - \sum_{j=0}^{i} \sigma(t_j) \Delta t \right) \]

let \( \alpha_j = 1 - \exp \left( -\sigma(t_j) \Delta t \right) \) \[ T_i = \prod_{j}^{i} (1 - \alpha_j) \]
The alpha blending view of volume rendering

first collect a bunch of samples (green dots)
next, do alpha blending to combine all their colors
Volume rendering is heavily used in medical imaging

Smoke effects in video games

https://twitter.com/CounterStrike/status/1638580074126659584
Modern 3D computer vision is heavily based on volume rendering

find the density and color that will render to observed images

https://alexyu.net/plenoxels/?s=09
Luma AI

https://lumalabs.ai/
So far, we ignore the fact that volumes can also reflect lights

absorptive volumes

scattering volumes
How to make volume react to light?

<table>
<thead>
<tr>
<th></th>
<th>σ = 0</th>
<th>σ = 1</th>
<th>σ = 1</th>
<th>σ = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>σ = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>σ = 10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>σ = 10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>σ = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>σ = 0</td>
<td>σ = 1</td>
<td>σ = 1</td>
<td>σ = 0</td>
<td></td>
</tr>
</tbody>
</table>
We need an equivalent to BRDF just like the surface case

\[ f(v, l) \]
We need an equivalent to BRDF just like the surface case

in volume rendering, \( f \) is usually called the “phase function”
A famous phase function: Rayleigh scattering

\[ f(v, l) = \frac{8\pi^4 \alpha^2}{\lambda^4} \left(1 + (v \cdot l)^2\right) \]

\(\alpha\): “polarizibility”

(https://en.wikipedia.org/wiki/Polarizability)

\(\lambda\): wavelength

Rayleigh scattering explains why sky is blue!

The strong wavelength dependence of Rayleigh scattering enhances the short wavelengths, giving us the blue sky.

The scattering at 400 nm is 9.4 times as great as that at 700 nm for equal incident intensity.
Simulating volumetric scattering

for each sample in raymarching, raymarch to the light as well

<table>
<thead>
<tr>
<th></th>
<th>$\sigma = 0$</th>
<th>$\sigma = 1$</th>
<th>$\sigma = 1$</th>
<th>$\sigma = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 0$</td>
</tr>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 10$</td>
<td>$\sigma = 0$</td>
</tr>
<tr>
<td>$\sigma = 0$</td>
<td>$\sigma = 1$</td>
<td>$\sigma = 1$</td>
<td>$\sigma = 1$</td>
<td>$\sigma = 0$</td>
</tr>
</tbody>
</table>

this is analogous to direct lighting in surface rendering
Simulating volumetric scattering

for each sample in raymarching, raymarch to the light as well

this is analogous to direct lighting in surface rendering
Volumetric path tracing
(CSE 272 HW2)
Volumetric path tracing
(CSE 272 HW2)
Volumetric path tracing
(CSE 272 HW2)

sample a distance
Volumetric path tracing
(CSE 272 HW2)

sample a distance

add emission
Volumetric path tracing
(CSE 272 HW2)
Volumetric path tracing
(CSE 272 HW2)
Volumetric path tracing
(CSE 272 HW2)
Volume scattering in visual effects
Volumetric scattering of skin
Next: neural networks!