More Volume Rendering Issues

Volume Rendering Techniques
- **Image Order**
  - Cast a ray from each image pixel and process each voxel along the ray
  - Disadvantage is that the spatial data structure must be traversed once for every ray, resulting in redundant processing of the volume data (the volume data is not accessed in storage order because of the arbitrary traversal direction of the viewing rays)
- **Object Order**
  - Maps (splat) the data samples to the image plane while processing through the volume data in storage order
  - Disadvantage is that early ray termination is difficult to implement
- **Hybrid**
  - First apply object-order technique followed by an image-order technique

VTK
- Vtk only supports vtkImage data for volume rendering
  - Isotropic – samples taken at regularly spaced intervals along all three axes
  - Anisotropic – spacing between samples along each axis is constant, but there may be three different spacing constants for the three axes

Volume Vs. Geometric Rendering
- translucent objects can be represented by surface methods
- Surface methods are inadequate for modeling and rendering amorphous phenomena (e.g., clouds, fire, smoke) that lack any tangible surfaces
- Cheap, powerful surface method acceleration hardware (polygon engines) is widely available
- Volume engines are emerging

Volume Visualization
- Discuss some tradeoffs along with additional techniques to address various issues including performance

Trade-offs in Image-Ordered Methods
- **Step Size**
  - Larger step size along the ray decreases the execution time of the ray function algorithm
  - Larger step sizes often lead to artifacts in the image
- **Sampling Method**
  - Uniform sampling across the volume gives a smooth result
  - Voxel-by-voxel traversal is much faster
- **Interpolation**
  - Can use linear interpolation across cell
  - Shortcut: simply use the nearest voxel value
Step Size Effect

Step size and interpolation method have less effect if the ray lies along an axis of the grid.

In general, some voxels can be missed due to sampling even if the spacing is 1 voxel width which can lead to artifacts.

Uniform Sampling Method

Represent ray as follows: \((x, y, z) = (x_0, y_0, z_0) + (a, b, c)t\)

in which \((x_0, y_0, z_0)\) is the ray origin, for example, the pixel for parallel perspective; and \((a, b, c)\) is the normalized ray direction vector.

Let \(t_1\) and \(t_2\) be the distances where the ray enters and exits the volume, respectively and let \(\delta_t\) be the step size the uniform sampling.

\[
\begin{align*}
    t &= t_1; \\
    v &= \text{undefined}; \\
    \text{while (} t < t_2 \text{)} \\
    \{ \\
    x &= x_0 + a \cdot t; \\
    y &= y_0 + b \cdot t; \\
    z &= z_0 + c \cdot t; \\
    v &= \text{EvaluateRayFunction}(v, t); \\
    t &= t + \delta_t \}
\end{align*}
\]

accumulate ray value over each iteration for compositing

Voxel Traversal Method with Nearest Neighbor- Faster

Examine each voxel along the ray rather taking samples. Use nearest-neighbor along with voxel path.

Templates

- Similar forms of digital parallel rays (inter-ray coherence) suggested using a pre-calculated template in ray casting

Templates for path of voxels

Rays in parallel viewing perform the same set of steps which is saved in a ray-template. Rays have to be traced from a base-plane.

Employ a templated ray casting technique with a discrete ray type to generate images.

Discrete Ray Types

Two voxels are 26-connected if they share either a vertex, an edge, or a face.

Edge-sharing and face-sharing voxels are defined as 18-connected.

Face-sharing voxels are defined as 6-connected.
**Voxel Traversal with Templates**

- Rays are parallel
- Each ray forms an identical path of voxels through the grid to form a template
- Save computation time
- Some voxels will be skipped resulting in artifacts

**Re-sample Approach**

- Start templated rays in each voxel at the edge of the grid
- Some voxels may be visited many times

**Shear-Warp Rendering**

- To get fast traversal, shear volume by translating each slice into the sheared space ... then can resample as shown
- Project front-to-back to get intermediate image
- Then warp image

**Shear-Warp Factorization**

- Hybrid Technique
- Instead of traversing along rays, visit voxels in a plane
  - Makes it possible to simultaneously traverse samples and pixels (traverse in data order and pixel order simultaneously)
  - Visit voxels in front-to-back order
  - Supports early termination
  - Need to perform a final 2D warp on the image due to the shear process
  - Can dramatically improve efficiency but can introduce blurring in some cases

**Transform to Sheared Space**

- Shear volume such that rays are perpendicular to the base plane
- Rays are cast from the base plane voxels at the same place
- They intersect voxels on subsequent planes in the same location
- Only one set of interpolation weights needs to be computed for all the voxels in a plane

**Camera Rotation**

- No rotation
- Small angle
- Large angle
Final Process of Shear-Warp

Shear-warp algorithm has 3 conceptual steps:
1. shear and resample the volume slices
2. project resampled voxels scanlines onto intermediate scanlines
3. wrap the intermediate image into the final image.

See Lacroute and Levoy paper for details.

Example of Shear Warp Rendering

- Re-sampling (e.g., during the final warp) can contribute to blurring and loss of detail
- Potential is greatest for degradation when view angle is 45 degrees with respect to slices of the data set and have high frequency transfer functions
- Artifacts can be reduced by using smoother transfer functions

New Hardware Advances

- Holy grail: real-time volume rendering
- Cube-4 Architecture SUNY Stony Brook was licensed by Mitsubishi
- VolumePro 500 system first commercially available from Mitsubishi’s RealTime Visualization (1999), now from TeraRecon VolumePro 1000 (2002)
- Uses (VolumePro 500) shear-warp rendering, tri-linear interpolation, central-difference gradient estimation, post-classification via a 4k X 36-bit lookup table for color and opacity values, Phong shading, etc.

Adaptive Screen Sampling

- Rays are emitted from a (sparse) subset of pixels
- Missing values are interpolated
- In areas of high value gradient additional rays are traced (refinement)

Adaptive Ray Sampling

- Sampling rate is adjusted to the significance of the traversed data
- Sampling rate as a function of gradient magnitude
- Sampling rate as a function of material opacity
Volume Illumination

• Why do we want to do it?
  • Potentially enhance display cues about the orientation of opacity gradients or embedded iso-surfaces within the volume
    – Surfaces facing the light will be brighter than surfaces facing away from the light
  • Can illuminate either an embedded surface or a sharp gradient in opacity (e.g., edge of a cloud)
  • Not always appropriate
    – If an area is dark, is it dark due to a shadow or is there really nothing there?

Volumetric Illumination

• Volumetric illumination is complicated. Some models consider the following:
  • absorption
  • emission
  • reflection
  • scattering (including multiple scattering)

Global Illumination of Volumes

• For every ray consider
  1. Light absorbed
  2. Light emitted
  3. Light scattered out of the ray
  4. Light scattered into the ray

• Scattering is often ignored in volumetric rendering
• Low-albedo model assumes light ray only scatters once before leaving the volume

Classical Approach - Volume Rendering Integral

• Cast rays through image plane into volume, and measure light received
• $C(s)$ is attenuated due to absorption and scattering
• $I(x, r)$ is the amount of light of a given wavelength coming from ray direction $r$ that is received at location $x$ on the image plane
• Must evaluate this equation for different wavelengths (e.g., r, g, b) separately

$L_s C(s) = \text{light reflected at point } s$:
includes emission, transmission, and reflection

$\mu(s) = \text{light extinction coefficient at location } s$ along ray

Think of volume being composed of particles of mass density $\mu$

$\mu(s)$ models the attenuation of light per unit length along the ray due to scattering or extinction

$C(s)$ models light added per unit length along the ray (at location $s$) in the direction $r$

$C(s) = E(s) + T(s) + R(s)$

self-emission transmission

from standard illumination equation with ambient, diffuse, specular components

$C(s) = E(s) + T(s) + R(s)$

attenuation factor that models the absorption of light along the ray by the mass densities of the particles along $r$.
Volume Rendering Approximation

- In general, the volume rendering integral cannot be computed efficiently, if at all, therefore a variety of approximations are used.

\[
I(x, r) = \sum_{n=0}^{\infty} C(n\Delta s) \alpha(n\Delta s) \Pi_{j=0}^{n-1}(1-\alpha(j\Delta s))
\]

- Approximate the integral with a sum and exp with a product
- Sample ray of length \(L\) at intervals \(\Delta s\) apart
- Let \(n = \lfloor L/\Delta s \rfloor\)

\[
I(x, r) = \sum_{i=0}^{n-1} C(i\Delta s) \mu(i\Delta s) \Delta s \prod_{j=0}^{i-1}(1-\exp(-\mu(j\Delta s)\Delta s))
\]

- Let transparency \(t(i\Delta s) = \exp(-\mu(i\Delta s)\Delta s)\)
- Approximate \(\exp(-x) \approx 1-x\) (2 term Taylor series)

\[
\mu(i\Delta s) \Delta s \approx 1 - t(i\Delta s) = \alpha(i\Delta s)
\]

\[
I(x, r) = \sum_{i=0}^{n-1} C(i\Delta s) \alpha(i\Delta s) \prod_{j=0}^{i-1}(1-\alpha(j\Delta s))
\]

- Let \(C(i\Delta s) = I_i\)
- Let \(\alpha(i\Delta s) = \alpha_i\)

\[
I(x, r) = I_0 \alpha_0 + I_1 \alpha_1 (1-\alpha_0) + I_2 \alpha_2 (1-\alpha_0)(1-\alpha_1) + \ldots
\]

Composition Methods

- Since the "over" operator is associative, but not commutative, we can group, but not reorder, to compute \(I(x, r)\) in different ways

Back-to-front: \(I(x, r) = (I_0 \alpha_0 \over (I_1 \alpha_1 \over (I_2 \alpha_2 \over \ldots)))\)
Front-to-back: \(I(x, r) = (((I_0 \alpha_0 \over I_1 \alpha_1 ) \over I_2 \alpha_2 ) \over \ldots)\)
Parallel: \(I(x, r) = ((I_0 \alpha_0 \over I_1 \alpha_1 ) \over (I_2 \alpha_2 \over \ldots))\)
• Calculate colors and opacities at each sampling site
• Weight these colors and opacities at each sampling site by the current accumulated transparency \((1 - \alpha)\)
• Add these terms to the accumulated color and transparency to form the terms for the next sample along the ray.

\[
\begin{align*}
\text{Intensity } I_0^* &= \alpha_0 I_0 \\
\text{Opacity } \alpha_0 \\
\text{Intensity } I_1^* &= I_0^* + (1 - \alpha_0) I_1 \\
\text{Opacity } \alpha_1 \\
\end{align*}
\]

Front-to-Back

### Shading

- The appearance of volume rendered images depends critically on the reflectance calculation used to shade the samples.

### Reflectance

\[
R(s) = k_a C_a + k_d C_i L(s) \cdot N(s) + k_s C_o H(s) \cdot N(s) \]

- \(C_a\) - ambient color
- \(C_i\) - color of light source
- \(C_o\) - object color
- \(L\) - light direction vector
- \(N\) - normal (determined by gradient)
- \(H\) - halfvector (Blinn-Phong Model)
- \(n_s\) - specular power

### Light to the eye

\[
I(x, r) = \int_0^R C(s)\mu(s)e^{-\int_s^r \mu(t)dt} ds
\]

Equation above only models attenuation of light from source to the eye.

\[
C_i(s) = C_i \exp\left(-\int_0^s \mu(t)dt\right)
\]

Inclusion of the term \(C_i(s)\) above into \(R(s)\) produces volumetric shadows which may provide greater realism to the image.

### VTK Implementation

- Defines ambient, diffuse, specular, and specular power as part of the \(\text{vtkVolumeProperty}\) (similar to how it associates these shading parameters with actors)

- If the classification is done at the vertices, there is a choice:
  - Gouraud-type: shade vertices and then interpolate to get reflected color at sample point
  - Phong-type: interpolate normals at vertices to get sample normal and then calculate shading

- Phong-type gives better quality - at the cost of more computational effort
Normal Vector

- If we plan to interpolate from vertices to get sample normal, then we need to store normals for entire dataset so shading calculations can be interactive.
- Naïve approach would store 1 floating point value (typically 4 bytes) per component of the gradient vector.
- For a 256 X 256 X 256 dataset of 1-byte scalars, storage would increase from 16 M to 218 M bytes (i.e., 13 bytes X 256 X 256 X 256).
- More “normals” resolution than typically needed.
- Solution is to quantize direction and magnitude into a smaller number of bits.

Storage of Normal Vector Direction

- Create a sphere and number the vertices.
- Encode the direction according to the nearest vertex that the vector passes through starting at the origin of the sphere.
- For infinite light sources, only need to calculate the shading values once and store these in a table.

Summary of Ray Casting Components

- Memory system provides the necessary voxel values at a rate which ultimately determines performance.
- Ray-path calculation determines the voxels that are penetrated by a given ray.
- Interpolation estimates the value at a re-sample location using a small neighborhood of values.
- Gradient calculations estimate a surface normal using a neighborhood of voxels.
- Classification maps interpolated sample values and the estimated surface normal to a color and opacity.
- Shading uses gradient and classification information to compute a color that takes into account the interaction of light on the estimated surfaces in the dataset.
- Composition uses shaded color values and opacity to compute a final pixel color for display.

Ray Casting Summary

- Advantages:
  - Non binary classification
  - Shows structure between surfaces
  - Can exploit parallel processing
- Disadvantages:
  - Computationally expensive - cost proportional to number of voxels (compositing is expensive).
  - Does not take advantage of triangle rendering hardware.

Ray Casting Examples

- The following sequence of slides were produced using the ray casting technique available in the vtk software.
- The slides show for the bonsai tree data set how different aspects can be highlighted by control of the opacity transfer function.
- The slides also show a comparison of different approaches to interpolation and shading.