ReVolVR – Rendering Volume Data in VR using HTC Vive

Sebastian Suder
suders Sebastian@alumni.fh-aachen.de

Stefan Schiffer
schiffer Stefan@fh-aachen.de

Ingrid Scholl
scholl Ingrid@fh-aachen.de

Mobile Autonomous Systems & Cognitive Robotics Institute (MASCOR)
FH Aachen University of Applied Sciences, Aachen, Germany

INTRODUCTION

Direct Volume Rendering (DVR) techniques are used to visualize surfaces from 3D volume data sets without computing a 3D geometry. Several surfaces can be classified using a transfer function (TF) by mapping data values to color and opacity (RGB). To find a good transfer function is a general manual and time-consuming procedure and requires detailed knowledge of the data and the imaging technique. In this poster, a new Virtual Reality (VR) application ReVolVR is presented. It is based on the HTC Vive VR technique to render and interact with volume data. ReVolVR loads, modifies, and saves the TF in real-time while continuously rendering the stereoscopic 3D volume through GPU-based ray casting shader.

RAY CASTING RENDERING METHOD

Ray Casting:
- Ray Casting is a direct volume rendering method.
- Ray Casting traces rays from the camera into the volume and uses sample values along the ray to compute a volume-rendering integral.
- GPU based ray casting parallelizes the computations for each ray [3, 1].

Ray Casting Improvements:
- Empty space skipping.
- Early ray termination: ray traversal can be stopped when the opacity ≥ 1.0.
- To avoid aliasing jitter the entry point from the ray through the fragment pixel.

Algorithm 1: Pseudocode for single pass ray casting with pre-integration

\[
\text{Algorithm 1: Pseudocode for single pass ray casting with pre-integration} \\
\begin{align*}
&\text{Empty space skipping.} \\
&\text{Apply electronic clipping to ray start position along ray direction.} \\
&\text{while} \quad (\text{Current ray position in volume}) \quad \text{AND} \quad (\text{opacity value on y-axis}) \\
&\quad \text{do} \\
&\quad \text{Calculate new transformation matrices with sensor tracked data.} \\
&\quad \text{Apply the transformations and render new stereoscopic image pairs.} \\
&\quad \text{Render three different clipping planes (transverse / sagittal / coronal planes) from the original voxel data.} \\
&\quad \text{Toggles on/off the clipping plane rendering.} \\
&\quad \text{Picking only one clipping plane with one controller.} \\
&\quad \text{Sliding the selected clipping plane by moving the picking controller.} \\
&\quad \text{The volume can be scaled by movement of both controller.} \\
&\quad \text{Toggles on/off the transfer function editor.} \\
&\quad \text{Load predefined transfer functions.} \\
&\quad \text{Toggle on/off the volume rendering.} \\
&\quad \text{The volume can be scaled by movement of both controller.} \\
&\quad \text{Transfer Function Editor:} \\
&\quad \text{Use our GUI design for the 1D transfer function (TF) editor (data range on x-axis, opacity value on y-axis) [4]. Mapping the GUI image as a texture into the virtual scene enables the following TF interactions:} \\
&\quad \text{Toggles on/off the transfer function editor.} \\
&\quad \text{Load predefined transfer functions.} \\
&\quad \text{Design interactively a new transfer function (set color and opacity ranges, define tent functions).} \\
&\quad \text{Render the volume in real-time with new visual properties in the virtual scene.} \\
&\quad \text{Save the current transfer function.} \\
&\text{END while} \\
\end{align*}
\]

VR CONTROL

1. Menu button
2. Trackpad
3. System button
4. Status light
5. USB charging adapter
6. Tracking sensor
7. Trigger
8. Grip button

VOLUME INTERACTION

Several user interactions are implemented to pick, translate and scale the volume data in the virtual scene. All interactions are designed to conveniently reflect to real movements of the controllers as much as possible.

Head and Controller Tracking:
- The Vive base station tracks sensors from the headset and controllers.
- Calculate new transformation matrices with sensor tracked data.
- Apply the transformations and render new stereoscopic image pairs.

Clipping planes:
- Render three different clipping planes (transverse / sagittal / coronal planes) from the original voxel data.
- Toggles on/off the clipping plane rendering.
- Picking only one clipping plane with one controller.
- Sliding the selected clipping plane by moving the picking controller.

Volume Transformations:
- Toggles on/off the volume rendering.
- Pick the volume through intersection of the volume with one controller.
- Translate the volume through movements of the picking controller.
- The volume can be scaled by movement of both controller.

Transfer Function Editor:
- We use our GUI design for the 1D transfer function (TF) editor (data range on x-axis, opacity value on y-axis) [4]. Mapping the GUI image as a texture into the virtual scene enables the following TF interactions:
- Toggles on/off the transfer function editor.
- Load predefined transfer functions.
- Design interactively a new transfer function (set color and opacity ranges, define tent functions).
- Render the volume in real-time with new visual properties in the virtual scene.
- Save the current transfer function.

Usability:
- In a preliminary evaluation run ReVolVR was received very well.
- Users felt it was very intuitive to use, even for technical novices.

Why we use the GPU?

Ray Casting Complexity Example:
- Volume Data Size 256³ voxel.
- HTC display resolution 2560 × 1080 pixel.
- Sample rate per ray < 300–400.
- Approx. 1.15 billion ops per user pose.
- With 24 fps we need over 27.6 billion ops.

Use GPU architecture:
- Implement all needed algorithms with shaders.
- Using shaders for the parallel raycasting.
- Real-time rendering on every interaction.
- Use high-performance computer with Nvidia GeForce GTX 1080.

REFERENCES