Content generation and real-time hologram computation for holographic 3D-displays

Enrico Zschau
SeeReal Technologies GmbH - Germany
Outline

Introduction to holography
Sub-Hologram – technology
Holographic Processing Pipeline on the GPU
Content generation
Hologram computation
Advanced content generation
Holographic 3D-display
Introduction to holography
### 3D-Stereo vs. Holography

- **3D-Stereo**
  - Mismatch between focus (display) and convergence (object)
  - Possible effects: headache, fatigue, short loss of orientation after longer use

- **Holography**
  - Like natural viewing – focus and convergence match
Conventional Holography

- A hologram is a large diffraction pattern
- 3D-scene is reconstructed when illuminated by coherent light – visible in the viewing zone – allows the eye to focus on different depths
- each hologram-pixel contributes to each scene-point
Conventional Holography

- Large viewing zone results from the very small pitch ➔ But most information calculated is wasted
- extreme resolution ➔ computation, memory, bandwidth
- Not in a foreseeable future esp. for real-time
Sub-Hologram - technology

- SeeReal’s Solution: Viewing Windows and Sub-Holograms
  - only the information the eyes can see is required ➔ small viewing zone – the “Viewing Window”
  - Viewing Window is shifted according to eye-positions using Eye-Tracking
  - larger pitch ➔ allows application of current LCD-technology
Sub-Hologram - technology

Sub-Holograms

- only a small area of a hologram which is responsible to reconstruct a scene-point needs to be calculated – a “Sub-Hologram”
- Direct computing of Sub-Holograms (Phases + Amplitudes) – no FFTs required
- Super-positioning of multiple Sub-Holograms
- Results in enormous savings regarding computation power
  ➔ Enables real-time computation using off-the-shelf components
For each scene-point to be reconstructed …
Sub-Hologram technology

Sub-Holograms are created and accumulated into the hologram.
Sub-Holograms for different scene-points …
Sub-Hologram technology

Sub-Holograms

… are super-positioned to create the hologram, reconstructing the complete scene inside the Viewing Window.
Holographic Processing Pipeline
Holographic Processing Pipeline

- Holographic-Software-System contains all steps to drive holographic displays

- **Holographic application**
- **Content-generator**
- **Color / depth maps and views**
- **Hologram-synthesis**
- **Complex-valued hologram**
- **Hologram-encoding**
- **Encoded hologram**
- **Post-processing**
- **hologram-frame**

- **SLM**
- **Light-Sources**

**Reconstructed 3D scene**

**Observer**

[Input from gesture detector / game-controller / mouse / keyboard]
For each observer and eye a color- and depth-map is required
Color/depth-maps represent a dense distribution of 3D-scene-points
Allows reconstruction of nice-looking holographic 3D-scenes
Eye-Tracking enables the natural Look-around effect of conventional holograms
Holographic Processing Pipeline

Hologram Synthesis

- Hologram Synthesis transforms object-points into complex valued Sub-Holograms
- Complex valued Hologram is created by superpositioning those Sub-Holograms
- Process is done separately for each view and color-component
- Hologram Synthesis, esp. the transformation from polar to cartesian coordinate system is most time consuming part in hologram calculation process

\[
\begin{align*}
\pi \star (x^2p_x^2 + y^2p_y^2) = \\
\lambda F + \Phi_0
\end{align*}
\]

\[
\begin{align*}
\text{RE} &= A \cos(\varphi) \\
\text{IM} &= A \sin(\varphi)
\end{align*}
\]

\[
A = \frac{b}{\sqrt{(SH_w \cdot SH_h)}}
\]
Holographic Processing Pipeline

Hologram Encoding

- Converts complex values to a representation compatible with spatial light modulator (SLM)
  - complex values cannot be written into a display ➔ change of representation necessary
  - i.e. 2 phases or 3 amplitudes (detour-phase / burckhardt encoding)
  - Requires normalization of amplitude

- Post processing step formats hologram data according to type of spatial light modulator
  - time-sequential / spatial presentation of color and views

Complex valued hologram → Complex Values (cartesian) → Transformation cartesian to polar coordinate system → Complex values (polar) → Conversion to SLM-compatible representation and Normalization → Pixel-values → Displayable hologram

\[ \phi = \text{atan2}(\text{IM}, \text{RE}) \]
\[ A = \sqrt{\text{RE}^2 + \text{IM}^2} \]
Hologram calculation using the GPU
Holographic Processing Pipeline

Hologram Calculation using the GPU

- Implementation for D3D 9 and D3D 11
- Hologram calculation implemented as a DLL, the Real Time Holography Library with very simple API
- An application is build on top of the RTH-Lib to enable holographic output on holographic 3D-displays
  - **no knowledge** about holography is necessary for the application designer – just draw your content
  - since all holographic processing is performed in the background **encapsulated** in the library
  - even correct setup of **virtual cameras** is handled automatically

- It is desirable, that this system could be directly implemented in the graphics driver to enable holographic output out of the box similar to Nvidias 3D-Vision
Holographic Processing Pipeline

Hologram Calculation using the GPU

- Multiple calculation stages making use of various vertex and pixel shader
- Intermediate results are stored in FP-rendertargets and used as FP-textures in the following stages
- Content is directly taken from application’s rendertarget (as color-texture + depth-texture)
  - Depth access: For DX9 Nvidia’s “INTZ” depth buffer is used, for DX11 this is straightforward
- Encoded hologram can then directly be written into a holographic display

3D-Application ➔ color-map ➔ Object-points as texture maps ➔ Generation of Sub-Holograms and accumulation ➔ Complex valued hologram ➔ Preparation For encoding ➔ Complex valued Hologram Prepared for encoding ➔ Find max. amplitude ➔ Complex valued Hologram with known normalization info ➔ Encoding and normalization ➔ A-map ➔ φ-map ➔ Hologram Encoding ➔ Hologram

Rendering of a specific view and layer ➔ Hologram synthesis ➔ RE-map ➔ IM-map ➔ Hologram Encoding preparation ➔ A-map ➔ φ-map ➔ Normalization ➔ A-map ➔ φ-map ➔ Max-map ➔ Hologram Encoding ➔ Hologram
For each object point a rectangle is drawn which represents a Sub-Hologram

- VS shifts vertices re Sub-Hologram size and position according to depth value
- Applies additional parameters to vertices for later use in PS using color/texture coordinate registers

Rectangle representing the formatted and parameterized Sub-hologram is handed over to rasterizer

\[ SH_{w,h} = \frac{|F| \lambda}{p^{2}_{x,y}} \]

\[ b = \frac{1}{\sqrt{(SH_{w} SH_{h})}} \]

\[ A = \frac{1}{\sqrt{(SH_{w} SH_{h})}} \]

- VS
- color-map
- depth-map
- Params-map (look up table)
- Random phase-map

For each object point a rectangle is drawn which represents a Sub-Hologram
Holographic Processing Pipeline

Hologram Synthesis using the GPU

- Rasterizer generates fragments from Sub-Hologram rectangle
- PS calculates $\varphi$ from interpolated parameters issued by VS
- transforms to cartesian space
- MRT is used to separately store real and imaginary values
- generated values are added into the hologram (RE / IM rendertargets)
- Adding is realized at no additional cost by using a special alpha blending mode

$\pi \times (x^2p^2_x + y^2p^2_y)$
$\varphi = \frac{\lambda F}{+ \varphi_0}$

$RE = A \cos(\varphi)$
$IM = A \sin(\varphi)$
Holographic Processing Pipeline

Hologram Encoding using the GPU - Preparation

- By drawing a fullscreen rectangle fragments for each hologram pixel are generated
- PS transforms hologram pixels from cartesian to polar space
- generated color-values are stored into the A/φ rendertargets

\[ \phi = \text{atan2}(\text{IM}, \text{RE}) \]
\[ A = \sqrt{\text{RE}^2 + \text{IM}^2} \]
Using a tree based reduction technique, maximum of all pixels in the whole hologram is calculated using only few rendering passes (< 8) with good shader-core utilization in comparison to full sequential maximum calculation.
Holographic Processing Pipeline

Hologram Encoding using the GPU - Normalization

- By drawing a fullscreen rectangle fragments for each pixel are generated
- PS calculates maximum of all pixels inside a NxM block in bound Amplitude-map
- generated maximum is stored in the maximum rendertarget

- For each NxM block in Amplitude-map one fragment is generated

Amplitude-map

Maximum calculation
PS

SV_Target0 (Maximum)
Holographic Processing Pipeline

Hologram Encoding using the GPU – Final step

- By drawing a fullscreen rectangle, fragments for each hologram pixel are generated.
- PS transforms complex values to SLM-compatible pixel-values.
- Performs normalization of amplitude using the maximum value.
- Generated color-values are stored in the hologram.

![Diagram of hologram encoding process](image-url)
Advanced content generation
Holographic Processing Pipeline

Content Generation – Transparency

- Sub-Holograms enable realization of (semi-) transparent objects
  - multiple scene-points in different depths at the same lateral position
- (Semi-) transparent scene-points stored in additional content-layers
- Additional alpha-map per layer defines grade of transparency

![Diagram showing layers of transparency](image)
(Semi-) transparent scene-points are realized …
… by creating a new one at the lateral position of existing scene-points …
… and accumulation/super-positioning into the hologram.

Two scene-points on the same ray: both generate a Sub-Hologram, are reconstructed and can be focussed.
Correct virtual camera setup when recording a scene is vital for a correct/natural reconstruction.

Recording a scene should be done using camera setup parameters …
Content Generation – Camera setup

- … mostly proportional to those parameters used for reconstructing the scene to enable 100% viewing comfort thus a natural representation

- **NO depth compression necessary** as used for 3D-Stereo, since unlimited depth range is possible at no discomfort
Holographic 3D-display Prototype
Current Holographic Prototype

20 inch direct view full color holographic 3D-display:

- **Off-the-shelf 20” / 60Hz** LCD with **7680x2048** amplitude modulating pixels
- Detour-phase / Burckhardt encoding

- Holographic encoding in vertical direction / **8mm Viewing-Window** at 2m distance
- Horizontal view multiplexing

- **Sequential color** presentation controlled by RGB-Backlight

- Thickness of housing comes from **large backlight and unoptimized optical components (low efficiency)** – final product designs will be much **thinner**
Current Holographic Prototype

- Currently one observer tracked by the integrated fast and precise Eye-Tracking (60 Hz, +/- 2.5 mm accuracy)

- PC using a single NVIDIA GTX 285 performs content generation and holographic calculation in real-time (60 Hz)

- Reconstruction of 640x512 scene-points with unlimited depth
- Multiple transparency depth cues per scene-point

- Resolution is not limited by the display, but by the eye’s capability to resolve more scene points at given display size and viewing distance
- Full HD resolution for large displays will be possible
Demo pictures taken with a camera

- Note: Pictures shown do not represent the same sharpness as perceived with human eye
Demo pictures taken with a camera

- Note: Pictures shown do not represent the same sharpness as perceived with human eye
Summary

- Conventional holography and Sub-Holograms
- Holographic processing pipeline
- Holographic calculation using the GPU
- Advanced content generation
- Holographic display
Outlook

Currently: Development of technologies for different display classes together with asian display-partners

- Head mounted display
- Mobile
- Desktop
- TV

FPGA-Version to integrate hologram calculation into display
⇒ later transition to ASIC

Further improvement and optimization of GPU-solution
⇒ CUDA 4/5
⇒ Compute Shader