

**Citation:**

Enrico Zschau, Robert Missbach, Alexander Schwerdtner, Hagen Stolle, "Generation, encoding and presentation of content on holographic displays in real time," Three-Dimensional Imaging, Visualization, and Display 2010, Bahram Javidi and Jung-Young Son, Editors, Proc. SPIE, Vol. 7690, 76900E (2010), doi:10.1117/12.851015.

**DOI Link:**

<http://dx.doi.org/10.1117/12.851015>

**Copyright:**

Copyright 2010 Society of Photo-Optical Instrumentation Engineers. One print or electronic copy may be made for personal use only. Systematic electronic or print reproduction and distribution, duplication of any material in this paper for a fee or for commercial purposes, or modification of the content of the paper are prohibited.

# Generation, encoding and presentation of content on holographic displays in real time

Enrico Zschau\*, Robert Missbach, Alexander Schwerdtner, Hagen Stolle  
SeeReal Technologies GmbH, Blasewitzer Str. 43, 01307 Dresden, Germany

## ABSTRACT

This paper discusses our solution for driving holographic displays with interactive or video content encoded in real-time by using SeeReal's *Sub-Hologram-technology* in combination with off-the-shelf-hardware. Guidelines for correctly creating complex content including aspects regarding transparency in holograms from both the content side and the holography side are presented. The conventional approaches for generating computer generated holograms are discussed in comparison with our solution using *Sub-Holograms*, to rapidly reduce computation power. Finally the computing-platform and the specification of our 20 inch direct-view holographic prototype will be presented.

**Keywords:** 3D-displays, Holographic displays, 3D-content, Real-time holography, Computer generated holograms, Sub-Holograms, Graphics processing unit, Field programmable gate array

## 1. INTRODUCTION

The conventional approaches to generate CGHs (computer generated holograms) are not well suited for interactive applications because of their massive consumption of computing power. So by using them, just still images or pre-calculated videos have been implemented. To realize the key benefits of 3D-holography, as compared to 3D-stereo, interactive content is essential – this provides a roadmap for combining typical 3D-applications as professional design, 3D-gaming or 3D-TV with the viewing comfort of 3D-holography. Accordingly, solutions for real-time holographic calculation without the need for high performance computing hardware are required.

This paper will present some background to create nice-looking interactive holographic applications. Furthermore the adaption of our novel Sub-hologram technology to effectively make use of graphics processing units or field programmable gate arrays will be discussed, which enables the calculation of holograms in real-time.

## 2. REAL-TIME HOLOGRAPHY

This chapter gives an overview about holography and especially compares the conventional approach against the novel Sub-Hologram technology from SeeReal, which is the basis to calculate large holograms in real-time.

### 2.1 Why holography?

Holography in comparison to 3D-stereo overcomes the problem of the depth-cue mismatch between depth-focus and convergence. This so called accommodation-convergence mismatch leads to fatigue or headache, even a short loss of orientation may occur, so with 3D-stereo, only small depth-ranges must be realized and the time to consume 3D-stereo without a break should also be very limited<sup>2</sup>.

Holography in contrast is like natural 3D-viewing, which allows very large depth ranges, there are no negative effects, because the eyes can both focus and converge on the object seen. When looking at a hologram, the object focused looks sharp while other objects in different distances will look blurry like it is in real life. In 3D-stereo the eyes converge on the object but focus on the display itself – a mismatch occurs, which leads to the effects already described above (see Figure 1a).

This is the reason, why holography will be the next big step in the currently rapid developing market for 3D-stereo, because it is the better option to many fields of applications, i.e. professional 3D-design, 3D-gaming and 3D-television.

\*E-mail: ez@seereal.com, Phone: +49 (0)351 450 3240, Web: <http://www.seereal.com>

The next section compares the conventional approach to create holograms with SeeReal’s novel solution, the so called Sub-Holograms. The use of Sub-Holograms enables to calculate large and deep holograms in real-time, which allows realizing interactive content on holographic displays using off-the-shelf hardware components.

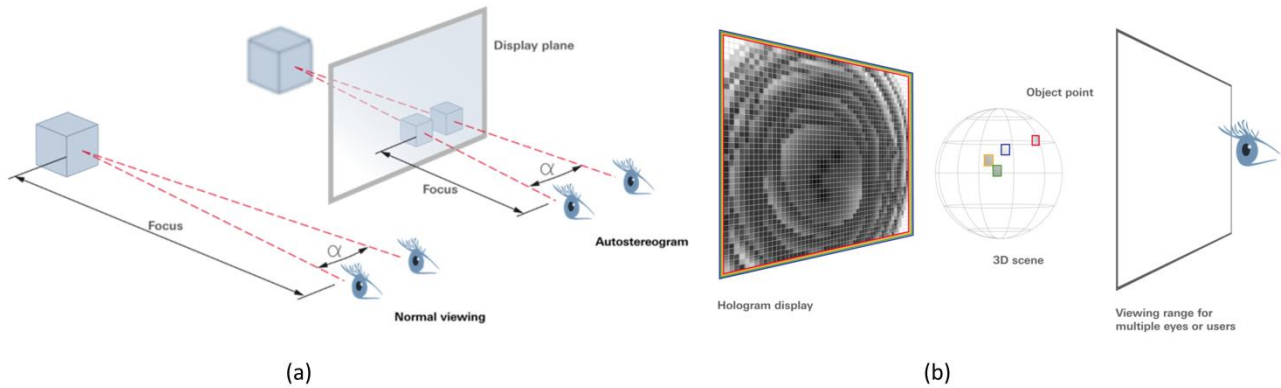


Figure 1: (a) The difference between 3D-stereo and natural viewing / holography: for 3D-stereo both eyes converge at the object in depth but focus on the display plane, for natural viewing and holography both focus and convergence are the same. (b) The principle of conventional holography: Multiple large overlapping diffraction patterns reconstruct multiple scene-points, when illuminated by a coherent light source – the reconstruction can be seen in a defined viewing-zone.

## 2.2 The conventional approach versus Sub-Holograms

A hologram is in general a complex diffraction pattern. When illuminated by a coherent light-source, a 3D-scene consisting of scene-points is reconstructed, which is viewable at a defined area in space (see Figure 1b).

The conventional approach to calculate computer generated holograms (CGHs) is generally based on the following scheme: Each pixel in a hologram contributes to each reconstructed scene point. That means, for each scene-point of a scene, a diffraction-pattern with the size of the full hologram has to be calculated. These individual holograms are all added up together – by complex superposition – to create the hologram representing the complete scene.

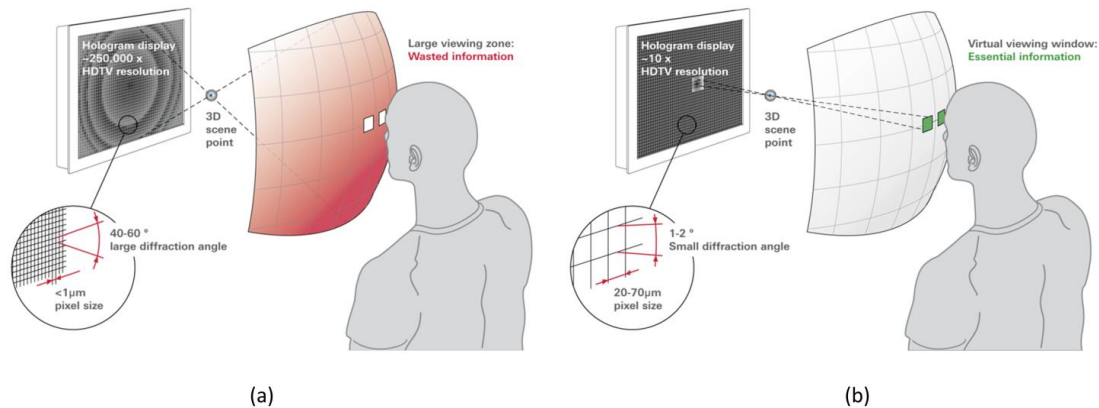


Figure 2: (a) When using the conventional approach, a large viewing-zone is generated, but only a small portion is really needed at the location of the observer’s eyes – so most of the calculated information is wasted. (b) Only the essential information is calculated when using Sub-Holograms. In addition the resolution of the holographic display is much lower and well within today’s manufacturing capabilities.

These conventional holograms provide a very large viewing-zone on the one hand, but need a very small pixel-pitch (i.e. around 1  $\mu\text{m}$ ) to be reconstructed on the other hand (see Figure 2a). The viewing-zone’s size is directly defined by the pixel-pitch because of the basic principle of holography, the interference of diffracted light. When the viewing-zone is large enough, both eyes automatically sense different perspectives, so they can focus and converge at the same point, even multiple users can independently look at the reconstruction of the 3D scene.

In a conventional hologram, the amount of pixels to calculate for each scene-point is immense. Beside the lack of a display-technology providing a small pitch at useful display-sizes, it would need incredible computing power. In addition the handling of such large amounts of data leads to even more problems regarding data transfer rate, memory and so on. This is a key reason why real-time holography using the conventional approach does not seem commercially viable in the foreseeable future. Because these of technical limits, only stills like hardcopies or chemical films could be realized in sizes appropriate for desktop or TV-like applications and with scalable technologies until now.

When looking at Figure 2b, it can be seen that most of the calculated information in a conventional hologram is wasted, because only the information the eyes can actually see is really needed. So instead of calculating the full viewing-zone, just that part of the hologram needs to be calculated, which is responsible for reconstructing a specific 3D scene point at the observer's eyes location – a Sub-Hologram (SH). This reduced viewing-zone is the so called Viewing-Window (VW) (see Figure 3).

The reduction of the size of this viewing-zone is done by increasing the pixel-pitch – the pixel-pitch along with other parameters defines the size of the viewing-zone. By overlapping (adding or super-positioning) the SHs of different scene-points, a holographic 3D scene with dense scene-points is reconstructed and visible at the location of the VW (see Figure 3).

The increased pixel-pitch on the other hand leads to a dramatically reduced pixel-count allowing the use of current display-technologies as another motivation. But the use of a small VW also implies the need of a fast, reliable and very precise Eye-Tracking system to shift the VW according to the observers' eye-movements. Such Eye-Tracking systems have already been developed, but currently SeeReal's uses its own Eye-Tracking solution integrated into holographic prototypes, which already have been demonstrated by SeeReal at public events like SID, FPD Yokohama and Finetec.

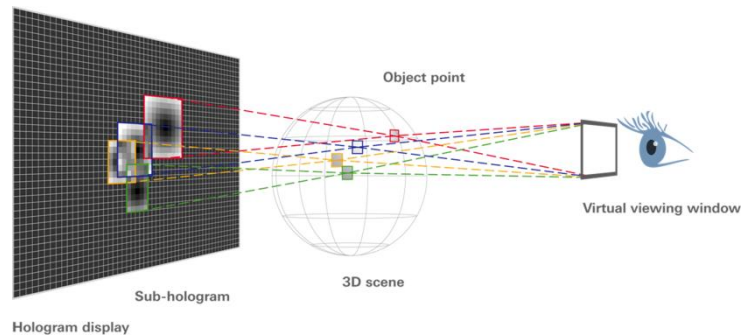


Figure 3: Only a small part of the hologram – a Sub-Hologram – is needed to reconstruct one single scene-point in the reduced viewing-zone – the Viewing-Window. By super-positioning multiple Sub-Holograms, a hologram representing the whole scene is generated and reconstructed at the Viewing-Window's location in space.

To give an example how enormous the savings of computing power are, both approaches have been compared in general using an exemplary situation.

Assuming to have a 40 inch SLM (800 mm x 600 mm), one observer is looking at the display from 2 meters distance, the viewing-zone will be +/- 10° in horizontal and vertical direction, the content is placed inside the range of 1m in front and unlimited distance behind the hologram, the hologram reconstructs a scene with HDTV-resolution (1920x1080 scene-points) and the wavelength is 500 nm, then the situation specified in Table 1 has to be managed.

Here for the conventional approach the calculation is based on Fourier-transforms<sup>1</sup> to apply the most efficient method for such large holograms, for this is assumed to have a depth quantization for the scene-points of 256 steps. For the SH-approach it is required to calculate two independent holograms, one for each eye.

At the bottom line, both approaches provide the same result for an individual observer position, but the significant difference in regards to resolution of the light modulator, frame-size and computing power can be clearly seen.

To further reduce the computation power, so called single-parallax holograms can be used, where the size of a SH and holographic parallax is reduced to one dimension. This is possible for vertical or horizontal direction – so called horizontal-parallax-only (HPO) or vertical-parallax-only (VPO) holograms<sup>3</sup>. By mixing half-parallax SHs with different views for each eye, for example a vertical holographic parallax, with a horizontal stereo-parallax, a real-time video-

hologram with low computational needs can be created<sup>8</sup>. The perceived limitations of single-parallax reconstructions are not visible to an observer if well understood and incorporated into holographic content.

Table 1: Comparison between the conventional and Sub-Hologram approach.

	Conventional Hologram	Hologram based on full-parallax Sub-Holograms and tracked Viewing-Windows
<i>SLM pixel-pitch</i>	1.4 $\mu\text{m}$	100 $\mu\text{m}$
<i>Viewing-Window / Viewing-zone</i>	700 mm x 700 mm / 700 mm x 700 mm	10 mm x 10 mm / > 700 mm x 700 mm
<i>Depth Quantisation</i>	256 steps	$\infty$
<i>Hologram-resolution in pixels</i>	$\sim 572\text{K} \times 429\text{K} \approx 246 \text{ GPixel}$	$8000 \times 6000 \approx 48 \text{ MPixel}$
<i>Memory for one hologram-frame (2x4 byte per hologram-pixel)</i>	1968 GByte	2 x 384 MByte (two holograms, one for each eye)
<i>Float-operations for one monochrome frame</i>	$\sim 33 \text{ PetaFlops}$ (by using an optimized FFT-based calculation)	2 x 182 GigaFlops (by using the direct Sub-Hologram calculation)

But even holograms providing full parallax SHs can be handled with SeeReal’s algorithms using today’s state-of-the-art technologies like field programmable gate arrays (FPGAs) and graphics processing units (GPUs), which provide sufficient computing power. This is being discussed in the following sections.

### 3. SEEReal’s HOLOGRAPHIC PROCESSING PIPELINE

The next four sections provide an overview of the important steps to showing real-time 3D-content on a holographic 3D display by using Sub-Holograms as explained above.

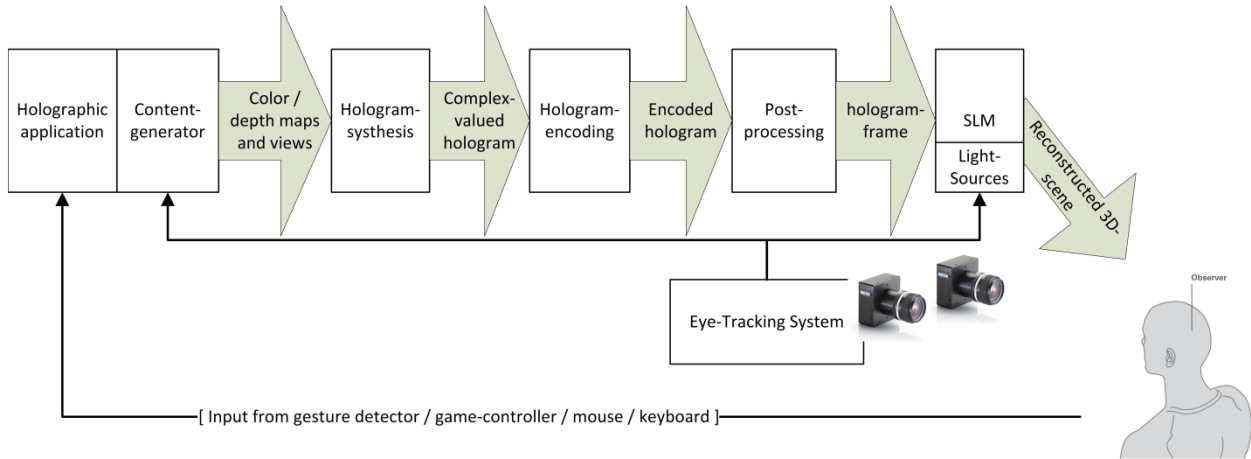


Figure 4: A general overview of our holographic processing pipeline.

The steps shown in Figure 4 define our holographic software pipeline, which is separated into the following modules: Beginning with the content creation, the data generated by the *content-generator* will be handed over to the *hologram-synthesis*, where the complex-valued hologram is calculated. Then the *hologram-encoding* converts the complex-valued hologram into the representation compatible to the used spatial light modulator (SLM), the holographic display. Finally the *post-processor* mixes the different holograms for the three color-components and two or more views dependent on the type of display, so that at the end the resulting frame can be presented on the SLM.

## 4. STEP I: CONTENT-GENERATION

For holographic displays, two main types of content can be differentiated. At first there is real-time computer-generated (CG) 3D-content like 3D-games and 3D-applications. Secondly there is real-life or life action video-content, which can be live-video from a 3D-camera, 3D-TV broadcast channels, 3D-video files, BluRay or other media.

For most real-time CG-content like 3D-games or 3D-applications, current 3D-rendering APIs utilizing graphics processing units (GPUs) are convenient. The most important ones are Microsoft's Direct3D and the OpenGL-API.

When creating and rendering a scene, for each view a 2D-map (a texture in terms of 3D-rendering API's) with pixels is created, where each pixel provides color along with its 2D-position. Each pixel can also be seen as a scene-point of the corresponding 3D-scene. This is the reason why both APIs are in general very suitable to generate content to be processed by SeeReal's holographic processing pipeline.

### 4.1 Views, color and depth-information

In SeeReal's approach, for each observer two views are created, one for each eye. The difference to 3D-stereo is the additional need of exact depth-information for each view – usually supplied in a so-called depth-map or z-map bound to the color-map. The two views for each observer are essential to provide the appropriate perspective view each eye expects to see. Together they provide the convergence-information. The depth information provided with each view's depth-map is used to reconstruct a scene-point at the proper depth, so that each 3D scene-point will be created at the exact position in space, thus providing a user's eye with the correct focus-information of a natural 3D scene. The views are reconstructed independently and according to user position and 3D scene inside different VWs, which in turn are placed at the eye-locations of each observer.

The provided depth-information has to be very precise because the depth of a scene-point given in the depth-map and its depth information provided by the parallax of the two views must correlate. This is essential to reconstruct a scene point at the right place in the viewing volume of the holographic display, so that focus and convergence will match. The depth-information is later used to create the correct diffraction pattern, the Sub-Hologram, which allows the eye to focus exactly at the convergence point.

### 4.2 Virtual cameras

Another important point which also applies to 3D-stereo, but is often underestimated by content creators, is the correct 3D-camera-setup (real cameras or virtual cameras for real-time 3D-content) from which the views for both eyes are taken.

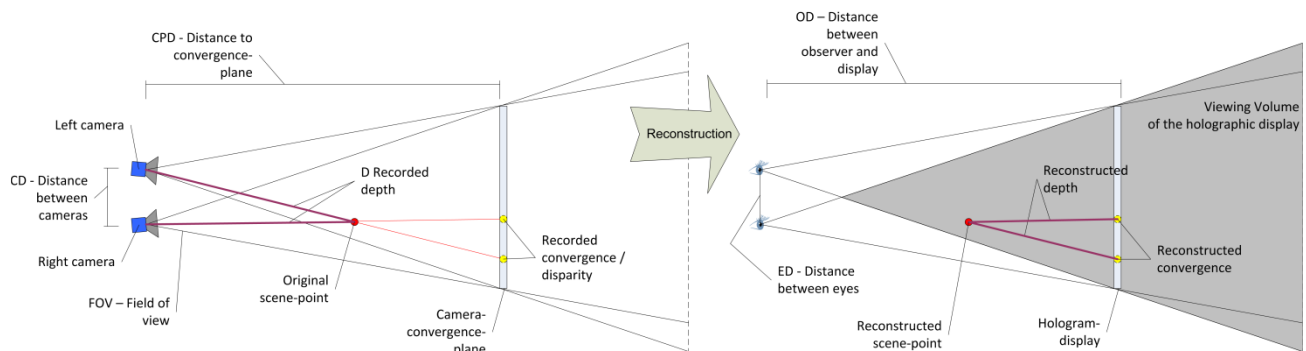


Figure 5: Schematic view of a camera-setup and the resulting reconstruction. If both FOV (field of view) and the relation between CD (camera distance) and CPD (convergence-plane distance) are nearly equal to the setup ED / OD (eye distance / observer distance), the holographic display provides, a 1:1 reconstruction can be achieved.

The (virtual) cameras, from where the convergence-information is recorded, should ideally have the same setup as the observers are sensing with their eyes. That means the cameras should be positioned at the locations of the eyes with convergence at the display-plane. Then an original scene could be recorded and would be 1:1 reconstructed. In general, the relation between the parameters for the camera-setup should be nearly the same as the setup the holographic display provides –  $CD/CPD$  should be nearly  $ED/OD$  (see Figure 5). The camera's field of view should provide nearly the same angular region, which is spanned from the display-plane to the observer's eye. Only by taking these restrictions into

account, a scene can be reconstructed and seen “as it would be really there”. Otherwise the scene would look similar, but with slight modified perspective or size, dependent on which parameters are not ideal. If the parameters are too different from the proper geometry, a strong perspective mismatch may occur.

These restrictions can be considered easily for real-time computer-generated 3D-content, because there the virtual cameras can be placed and modeled freely in the virtual space. Such virtual cameras are typically represented by so called view- and projection-matrices in terms of 3D-rendering APIs. For real-world (3D-camera) or offline computer-generated content (i.e. computer animated movies), the above mentioned restrictions should be kept in mind, because once the images are taken, the camera-setup could not be changed afterwards. A compromise could be the automatic generation of all required views from one central perspective, containing color and depth-information, from which all other perspectives are generated<sup>10</sup>. But there a loss of quality has to be taken into account, mainly because of missing occlusion-information. Since only one perspective is available, important perspective information, objects are occluding in the central view, is not available in other perspectives. Nonetheless, embodiments are possible to include occlusion data as part of the data stream.

For real-time computer-generated 3D-content, the actual observer-position in front of a holographic display with user tracking can be used for proper positioning of the virtual cameras. In SeeReal’s holographic 3D displays with information only within viewing windows, knowledge of eye coordinates can also be used for positioning the virtual cameras corresponding to position and movement of the observers, such providing the full viewing range (“look around” effect) of a natural scene. That means, the scene seems to be fixed at the same place, like in nature, when looking around a stationary object. In addition to providing all natural depth cues for comfortable long-time 3D viewing, this creates the same large viewing-zone, a conventional ultra high resolution hologram would inherently provide. Furthermore this feature is used to properly set the distance of the virtual cameras according to the observer’s eye-distances to perfect the virtual camera-setup. The look-around effect is a voluntary (switch on/off) feature though, which will not be desirable for all 3D content.

### 4.3 Transparency

An interesting effect, which is a unique feature of SeeReal’s holographic processing pipeline, is the reconstruction of scenes including (semi-) transparent objects. Transparent objects in the nature, like glass or smoke, influence the light coming from a light-source regarding intensity, direction or wavelength. In nature, eyes can focus both on the transparent object or the objects behind, which may also be transparent objects in their parts.

Such a reconstruction can be achieved straightforward using SeeReal’s solution to holography and has been realized in display demonstrators the following way: Multiple scene-points placed in different depths along one eye-display-ray can be reconstructed simultaneously. This means super-positioning multiple SHs for 3D scene points with different depths and colors at the same location in the hologram and allowing the eye to focus on the different scene-points at their individual depths (see Figure 6a). The scene-point in focal distance to the observer’s eye will look sharp, while the others behind or in front will be blurred. Unfortunately a reconstructed 3D scene point cannot realize the physical behavior a transparent object in nature will perform with passing light waves. So this behavior has to be simulated by manipulating the colors of the scene-points accordingly to realize effects like color-filtering or damping.

From the side of content-creation this transparency-effect can be controlled by adding an alpha-value to each scene-point, beside the existing color and depth-value. If a scene-point has an alpha-value of 0.0 (totally transparent) it will not be reconstructed, no SH will be created. An alpha-value of 1.0 means the scene-point is totally opaque, for this a single SH will be created – all scene-points behind will not be visible and no SH will be created for them. A value between 0.0 and 1.0 means the scene-point is partly-transparent, while the alpha-value represents its grade of transparency, so for both the transparent scene-point and the ones behind or in front, Sub-Holograms will be created.

Current real-time 3D-Rendering APIs like Direct3D and OpenGL provide only one depth-value per pixel, because only one color map and one depth-map is typically used at the same time to store a rendered scene. When rendering transparency-effects, typically multiple passes are done by blending all transparent objects in their depth order against the others already rendered into the color-map. For these blending-passes, the generation of depth-values is typically discarded. The final depth-value of each pixel in the depth-map normally corresponds to the pixel behind all transparent objects. Therefore a solution was developed by SeeReal to use these state-of-the-art 3D-Rendering API’s on one hand and create depth-values for all transparent objects on the other hand.

SeeReal's principle of generating multiple 3D scene-points at the same position in the hologram-plane but with different depths is based on the use of multiple content data layers (see Figure 6b). Each layer contains scene-points with individual color, depth and alpha-information. These layers can be seen as ordered depth-layers, where each layer contains one or more objects with or without transparency. The required total number of layers corresponds to the maximal number of overlapping transparent 3D scene points in a 3D scene. This scheme is compatible with the approach to creating transparency-effects for 2D and stereoscopic 3D displays. The difference on one hand is to direct the results of the blending passes to the appropriate layer's color-map instead of overwriting the existing color. On the other hand, the generated depth-values are stored in the layer's depth-map instead of discarding them.

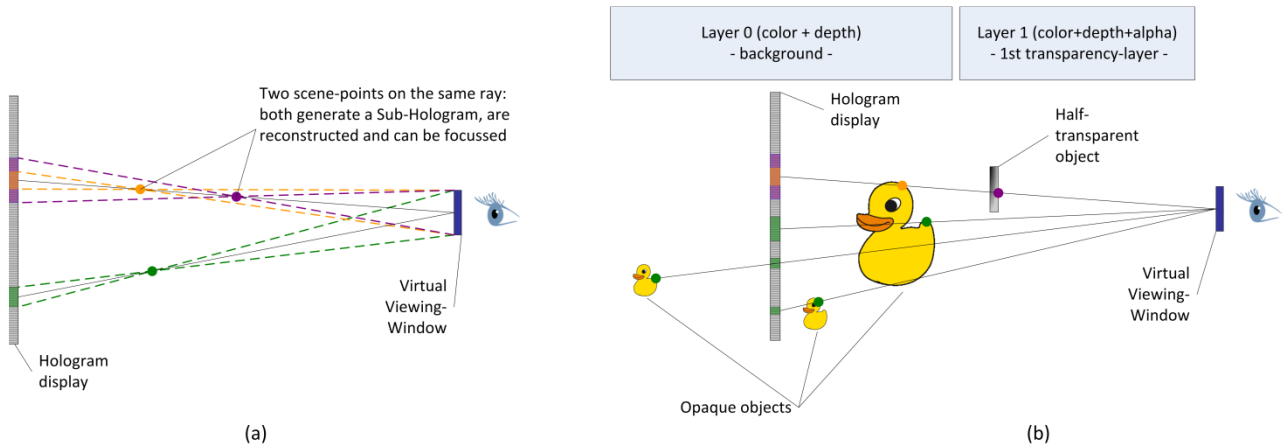


Figure 6: (a) Multiple scene-points along one single eye-display-ray are reconstructed consecutively and can be used to enable transparency-effects. (b) Exemplary scene with one additional layer to handle transparent scene-points (more layers are possible). Transparent scene-points are stored in the first layer, the background-objects reside in the background-layer.

Finally, the layers have to be preprocessed to convert the colors of all scene-points according to their given alpha-values and influences from other scene-points behind. As an example there are two objects, one 100% red and opaque in background, one 100% white and half-transparent (50% transparency,  $\alpha = 0.5$ ) in foreground, which just damps the light by 50%. After processing, the background object is damped – its new color is 50% red, the foreground-object is only 50% opaque, so its final color will be 50% white. When looking at such a reconstruction, the background-object will be darker, when occluded by the half-transparent white object in foreground, but both can be seen and focused.

So at the end after processing the alpha-values, the data handed over to the *hologram-synthesis* contains multiple views with multiple layers, each containing scene-points with just color and depth-values. Later, SHs will be created only for valid scene-points – only the parts of the transparency-layers actively used will be processed.

#### 4.4 A holographic video-format

There are two ways to play a holographic video: By directly loading and presenting already calculated holograms or by loading the raw scene-points and calculating the hologram in real-time.

The first option has one big disadvantage: The data of the hologram-frames must not be manipulated by compression methods like video-codec's, only lossless methods are suitable. Through the very random nature of holographic data, lossless compression technologies are not effective in significantly reducing the data-volume to achieve streaming from a hard-drive or optical media, not to mention streaming over IP-networks.

To overcome this, SeeReal proposes to use the original scene-points stored inside the different views / layers as stated above. This in combination with SeeReal's real-time hologram calculation enables to use state-of-the-art video-compression technologies like H.264 or MPEG-4, which are more or less lossy dependent on the used bitrate, but provide excellent compression rates. The losses have to be strictly controlled especially regarding the depth-information, which directly influences the quality of reconstruction. But when choosing high bitrates, even then a compression-rate of around 1:10 with minimal but very acceptable losses is possible.

SeeReal developed and uses a simple video-frame format storing all important data to reconstruct a video-frame including color and transparency on a holographic display. This flexible format contains all necessary views and layers per view, to store colors, alpha-values and depth-values as sub-frames placed in the video-frame (see Figure 7). Additional meta-information, stored in an xml-document or embedded into the video-container, contains the layout and parameters of the video-frames the holographic video-player needs for creating the appropriate hologram. This information for instance describes which types of sub-frames are embedded, their location and the original camera-setup, especially how to interpret the stored depth-values for mapping them into the 3D-coordinate-system of the holographic display.

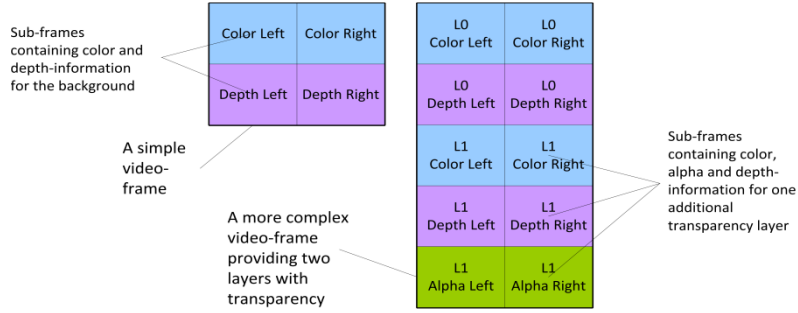


Figure 7: Typical example-layouts for video-frames. Sub-frames can be placed freely in the video-frame – a separate meta-information record provides the information, which sub-frames are embedded and their location in the video-frame.

This approach enables SeeReal to reconstruct 3D color-video with transparency-effects on holographic displays in real-time. The meta-information provides all parameters the player needs to create the hologram. It also ensures the video is compatible with the camera-setup and verifies the completeness of the 3D scene information (i.e. depth must be available).

## 5. STEP 2: HOLOGRAM-SYNTHESIS

The *hologram-synthesis* performs the transformation of multiple scene-points into a hologram, where each scene-point is characterized by color, lateral position and depth. This process is done for each view and color-component independently while iterating over all available layers – separate holograms are calculated for each view and each color-component.

For each scene-point inside the available layers, a Sub-Hologram  $SH$  is calculated and accumulated onto the hologram  $H$ , which consists of complex values for each hologram-pixel – a so called hologram-cell or cell. Only visible scene-points with an intensity / brightness  $b$  of  $b > 0$  are transformed, this saves computing time, especially for the transparency layers which are often only partially filled.

It is assumed the SLM provides the horizontal / vertical pitch  $p_x / p_y$ , the observer-distance between the hologram-plane and the observer is  $od$  and each scene-point provides its position  $(ox, oy)$  in the layer (which is also the Sub-Hologram-position in the hologram), its depth  $d$  and brightness  $b$  at the wavelength  $\lambda$  (according to the currently processed color-component).

At first the size of the Sub-Hologram  $SH_{w,h} = |F| \frac{\lambda}{p_x^2, p_y^2}$  in the display-plane (the hologram-plane) is calculated. Then for each cell  $SH(x, y)$  at the discrete cell-position  $(x, y)$  inside the SH, the complex value  $SH(x, y) = Ae^{-j\varphi(x, y)}$  is computed, the phase  $\varphi(x, y)$  and amplitude  $A$  are given by

$$\varphi(x, y) = \frac{\pi}{\lambda F} (x^2 p_x^2 + y^2 p_y^2) + \varphi_0 \text{ and } A = \frac{b}{\sqrt{SH_w SH_h}}$$

Such a SH describes a holographic lens with the focal length  $F = \frac{od d}{od - d}$  to reconstruct the given scene-point at the encoded distance  $d$ . The term  $\varphi_0$  ( $0 \leq \varphi_0 \leq 2\pi$ ) is an initial phase unique for each scene-point, which is typically random. The calculated SH is finally added up to the hologram  $H$  at the SHs location  $(ox, oy)$  in the hologram  $H$ . The position  $(ox, oy)$  is the center-position of the Sub-Hologram  $SH$  in the hologram  $H$ , which is defined by the ray crossing

the hologram / display-plane, starting in the VW and passing the scene-point's position in space and corresponds to the scene-point's 2D-position in the layer. Now  $H$  represents the wave-front which will reconstruct our 3D-scene.

In the next steps, the transformation of complex values to displayable real-values, the *hologram-encoding*, has to be performed to enable writing the hologram into the SLM.

## 6. STEP 3: HOLOGRAM-ENCODING

Encoding is the process to prepare a hologram to be written into a SLM, the holographic display. SLMs normally cannot directly display complex values, that means they cannot modulate and phase-shift a light-wave in one single pixel the same time. But by combining amplitude-modulating and phase-modulating displays, the modulation of coherent light-waves can be realized. The modulation of each SLM-pixel is controlled by the complex values (cells) in a hologram. By illuminating the SLM with coherent light, the wave-front of the synthesized scene is generated at the hologram-plane, which then propagates into the VW to reconstruct the scene.

Different types of SLM can be used for generating holograms, some examples are: SLM with amplitude-only-modulation (detour-phase modulation) using i.e. three amplitude values for creating one complex value<sup>4</sup>, SLM with phase-only-modulation by combining i.e. two phases<sup>5</sup> or SLM combining amplitude and phase-modulation by combining one amplitude- and one phase-pixel. Latter could be realized by a sandwich of a phase and an amplitude-panel<sup>6</sup>.

So, dependent of the SLM-type, a phase-amplitude, phase-only or amplitude-only representation of our hologram is required. Each cell in a hologram has to be converted into the appropriate representation. After writing the converted hologram into the SLM, each SLM-pixel modulates the passing light-wave by its phase and amplitude.

## 7. STEP 4: POST-PROCESSING

The last step in the processing chain performs the mixing of the different holograms for the color-components and views and presents the hologram-frames to the observers. There are different methods to present colors and views on a holographic display.

One way would be the complete time sequential presentation (a total time-multiplexing). Here all colors and views are presented one after another even for the different observers in a multi-user system. By controlling the placement of the VWs at the right position synchronously with the currently presented view and by switching the appropriate light-source for  $\lambda$  at the right time according to the currently shown hologram encoded for  $\lambda$ , all observers are able to see the reconstruction of the 3D scene. Another way is the independent presentation of the views and time-multiplexing of colors with two different SLM, such a system has already been shown by SeeReal at SID Long Beach in 2007<sup>7</sup>.

Two further methods, implemented in our current prototypes, are based on mixing colors and views in one single frame, or on mixing views in one single frame and presenting colors sequentially<sup>8</sup>. For both, single-parallax holograms in vertical direction (VPO) are used, while in horizontal direction the different views (and colors) are multiplexed using vertical interlacing.

There are also many other ways and methods to represent the colors and views for all observers. In general, the *post-processing* is responsible to format the holographic frames to be presented to the observers.

## 8. IMPLEMENTATION

All the processing-steps described above have been implemented as general software-modules inside SeeReal's holographic reference software-system. For the different prototypes the required code-paths have been optimized for GPU- and FPGA-computing to be running on off-the-shelf PC-hardware and on SeeReal's dedicated FPGA-platform. Both platforms are presented below but at first the specification of our 20 inch direct view holographic prototype will be given.

### 8.1 SeeReal's direct view holographic prototype

SeeReal's 20 inch direct view prototype is a full color holographic 3D display using 1D vertical encoding (VPO). It provides viewing-window tracking by using the integrated accurate and fast eye-tracking system, which delivers the 3D-eye-positions of up to four observers 60 times per second with an accuracy of +/-2.5 mm. But present holographic

prototypes are designed for one observer only. The eye-tracking software is running either on a standard PC or fully integrated into the display using an embedded DSP/FPGA-solution.

The holographic panel has a resolution of 2048x7680 amplitude modulating pixels and is running with 60Hz. The observer-distance is currently set at approximately 2m due to the (for holography) relatively coarse pixel pitch of the off-the-shelf LCD. By using detour-phase encoding<sup>4</sup>, the pitch for one complex pixel (cell) is 156x156  $\mu\text{m}$  – this leads to the corresponding VW of 8mm in vertical direction. In horizontal direction the holograms for both eyes are spatially separated. A fixed optical system creates horizontal sweet spots of 32.5 mm width for each eye, which are shifted in horizontal direction according to the observer's movements. So an observer can freely move the head in horizontal and vertical direction without losing the holographic view. The holograms for left and right eye are spatially multiplexed into the displayed frame while colors are sequentially presented. For each color-component (red, green, blue) a different frame for the appropriate wavelength is shown.

It is possible to reconstruct 512 x 2560 scene-points per layer, but at the observer-distance of 2m, a human eye cannot resolve a 3D scene resolution of 512 x 2560 on a 20 inch display. Accordingly, the 3D scene resolution is arbitrarily limited to 512 x 640 scene points per layer – this also provides a more common format (3:4). For a larger LCD, the 3D scene resolution can easily be scaled to Full-HD or higher. The depth-range of 3D scenes usually begins approx. 1 meter in front of the hologram display and can go to infinity behind the display.

## 8.2 Real-time holography on a PC

Motivation for using a PC to drive a holographic display is manifold: A standard graphics-boards to drive the SLMs using DVI can be used, which supports the large resolutions needed. Furthermore a variety of off-the-shelf components are available, which get continuously improved at rapid pace. The creation of real-time 3D-content is easy to handle using the widely established 3D-rendering APIs OpenGL and Direct3D on the Microsoft Windows platform. In addition useful SDKs and software libraries providing formats and codec's for 3D-models and videos are provided and are easily accessible.

When using a PC for intense holographic calculations, the main processor is mostly not sufficiently powerful. Even the most up-to-date CPUs, do not perform calculations of high-resolution real-time 3D holograms fast enough, i.e. an Intel Core i7 achieves around 50 GFlop/s<sup>9</sup>. So it is obvious to use more powerful components – the most interesting are graphics processing units (GPUs) because of their huge memory bandwidth and great processing power, despite some overhead and inflexibilities. As an advantage, their programmability, flexibility and processing power have been clearly improved over the last years.

## 8.3 Real-time holography using GPUs

Based on Microsoft's Direct3D 9, the complete holographic processing pipeline as presented above has been implemented along with some nice-looking interactive applications. Everything runs on a PC with one NVIDIA GeForce 285 GTX, driving our 20" holographic 3D direct view prototype. Almost all calculations are done on the graphics processing unit (GPU), the CPU is only used to control the program flow and to supply parameters for calculations. For most of the steps, special pixel- and vertex-shader-programs<sup>11</sup> have been implemented, which are switched from step to step to perform the appropriate algorithm. Shader-programs are small code-fragments written in a C-like language (i.e. HLSL when using Direct3D). When compiled and uploaded to the GPU for execution, they run in parallel inside the shader-cores of the GPU to process vertices or fragments/pixels inside the graphics-pipeline. A modern GPU has typically more than 200 shader-cores, with each one capable to perform typical 4-dimensional vector-operations at a high frequency at around 1 GHz.

The reason why the direct use of Direct3D towards other GPGPU-techniques (general purpose computation on graphics processing units<sup>13</sup>) like CUDA<sup>12</sup> was chosen is the greater flexibility to use any feature a GPU provides and the straightforward interface to the 3D-content created in real-time by the application-module.

Our solution, utilizing the GPU is very flexible and parameterizable. A Windows-SDK has been developed by SeeReal providing a simple API encapsulating all aspects of holographic computing, so any application-designer just needs to concentrate on the content itself. All things related to virtual cameras, hologram-synthesis, hologram-encoding and post-processing for a specific holographic display are hidden and automatically handled by the software-system. Figure 8 gives an overview of the GPU-based processing pipeline.

The GPU-solution used for SeeReal's 20" direct view prototype works as follows: In the first module, the *content-creation* – part of the holographic application – two views are created, each consisting of up to four layers, with each layer storing 512x640 scene-points and each scene-point providing color, depth and alpha-information. The holographic application may use all functions and features the Direct3D-API provides, such as its own shader-programs for example. The data generated – up to 16 textures in GPU-memory, two for each layer providing depth and color with alpha – is handed over to the next instance. This is the only module an application-designer must create and implement, all other parts are provided by the SDK.

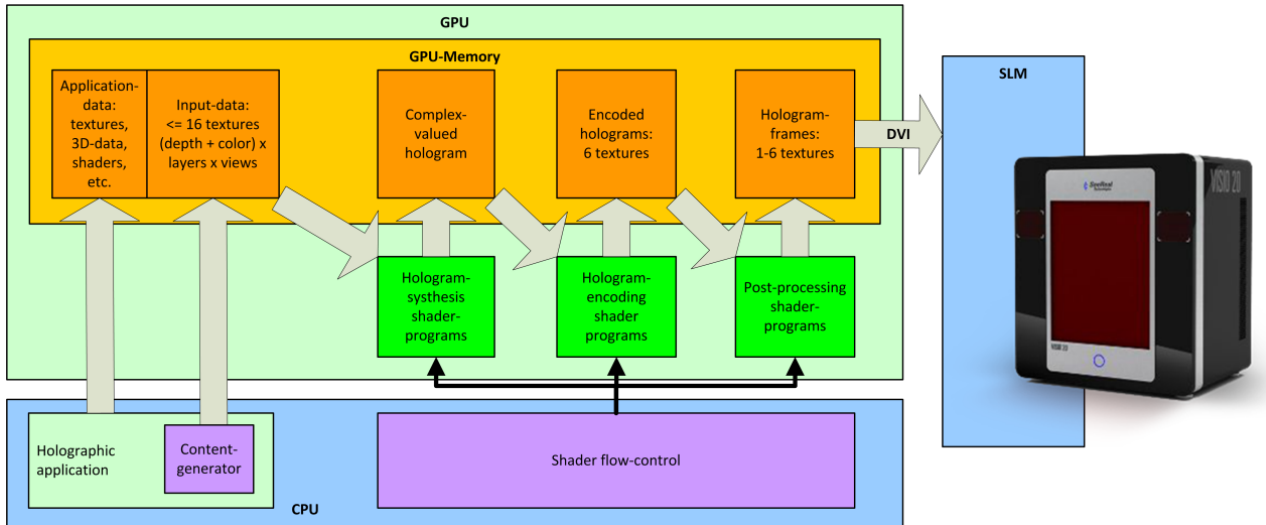


Figure 8: An overview of the data-flow in our GPU-solution.

The next module, the *hologram-synthesis*, processes each layer to apply the transparency-effects given by the alpha-value and performs some depth-sorting for the case that the depths between different layers have not been sorted by the application. Then for each 3D scene-point in each layer, a Sub-Hologram is generated and positioned using a vertex-shader-program. Each Sub-Hologram is processed by a pixel-shader-program performing the calculation of phases and amplitudes for each hologram-cell (complex value) as described above. Each calculated cell is then accumulated onto the resulting complex-valued hologram. This complex-valued hologram is implemented using two floating-point textures in GPU-memory, one for the real part, one for the imaginary part.

In our first prototype versions presented at SID in 2007, a lookup-table approach was used, where standardized Sub-Holograms had been pre-calculated for each of 256 discrete depth-steps. This was done to overcome the limited processing power of the GPUs at this time – i.e. the NVIDIA 7900 GTX. Until now, graphics processing units have been improved rapidly, but more in the direction of calculation power than on the memory-bandwidth. Now the combination of improved GPUs and optimized SeeReal algorithms enables direct computing of Sub-Holograms with greater quality, flexibility and efficiency – the dynamic Sub-Hologram size and direct computation leads to more efficient calculations, the nearly unlimited depth-resolution provided by the GPU (typically 24 bit) is now used to provide a finer depth-resolution in the reconstruction (esp. for large depth-ranges) and memory bandwidth does no more limit the calculations.

In the third step *hologram-encoding*, the complex values are encoded to create the final, SLM-compatible representation using detour-phase modulation<sup>4</sup>, which is also done using another set of pixel-shader-programs.

Finally the six holograms, two views with three holograms (for the color-components) each, are multiplexed for presentation on the SLM. Three SLM-frames, one for each color-component are created by multiplexing two views for each color-component into one frame using vertical interlacing. Then these three frames are sequentially displayed.

#### 8.4 Real-time holography using an FPGA

Another step in SeeReal's development of holographic 3D solutions was optimizing the software for porting to a field programmable gate array (FPGA). The motivation was to realize an autonomous system, to be integrated into any display. For this, a custom FPGA-board utilizing a Stratix III FPGA from Altera has been designed. The advantage of FPGAs against CPUs or GPUs is the mixing of best-of-both-worlds – much better parallelism than in CPUs combined

with greater programming flexibility than in GPUs. Nonetheless, the complex programming model of FPGA solutions leads to longer development cycles compared to PC-based developments.

On the application-side both solutions use a PC to create the content, using the *content-creation-module* like already described above. But for this solution, the data generated is transferred to the FPGA-board using DVI-Frames instead of textures inside GPU-memory, by packing the color and depth-information for both views inside one single DVI-frame. So a set-top box or gaming console would also be appropriate to use as content-source. Handling of more than one transparency layer will be realized by expanding the DVI-frame-size to include the information for up to four layers similar to the proposed video-frame format described above.

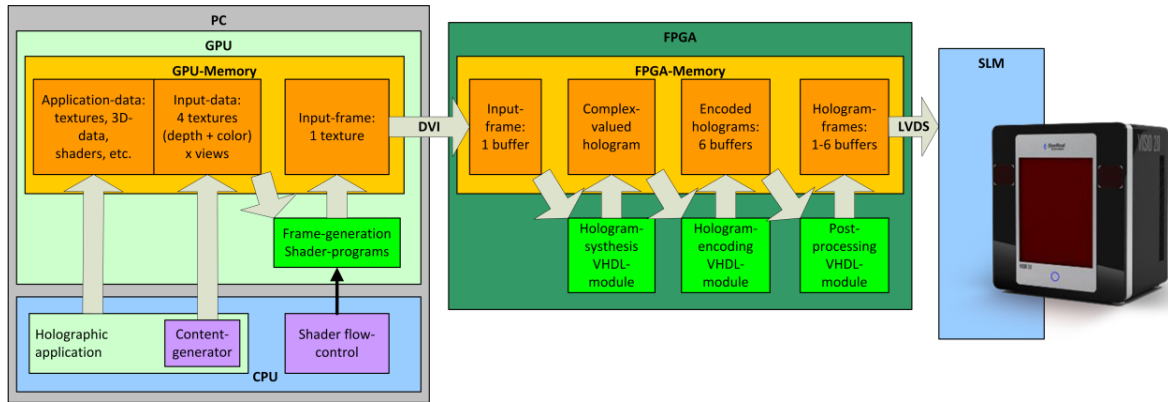


Figure 9: An overview of the data-flow in our FPGA-solution.

Modules for calculating Sub-Holograms, accumulation and encoding, as well as post-processing have been implemented using VHDL. Today this solution provides the same functionality as SeeReal’s GPU-version, except the pending support for layers. The FPGA-board directly drives the SLM using the LVDS-interface for presenting the frames (see Figure 9).

### 8.5 Applications

With those solutions available on GPU as well as FPGA, SeeReal is able to show live-encoded high resolution life action 3D-video with full display-frame-rate of 60 Hz, streamed frame by frame from a PC’s hard-drive using a standard MPEG-4 codec for video-decoding. Each frame contains raw color, depth and alpha-information for two views and 4 layers – background plus 3 transparency-layers – which are heavily utilized. More than 80% of all available scene-points are used in the transparency-layers for meaningful load comparisons.

Furthermore real-time applications have been developed and demonstrated showing computer-generated 3D-models based on standard formats and complex interactive 3D-scenes consisting of multiple models in a detailed 3D-environment. For enhancing the user experience and to simplify the interactions, selected modern human-machine-interfaces like a 3D-space-mouse and the Nintendo Wii controller have been integrated.

## 9. RESULTS

Based on SeeReal’s principles for holographic 3D displays and corresponding proprietary algorithms, both solutions GPU and FPGA are capable of driving SeeReal’s 20 inch holographic 3D direct view prototype with the full SLM-frequency of 60Hz using spatial view and sequential color multiplexing. Furthermore and already today, the solution is capable of driving scaled up display hardware (higher SLM resolution, larger size) with the correspondingly increased pixel quantity.

The high-resolution, full frame rate GPU-solution runs on a PC with a single NVIDIA GTX 285, SeeReal’s FPGA-solution uses one Altera Stratix III to perform all the holographic calculations. Transparency-effects inside complex 3D scenes and 3D-videos are supported. Even for complex high-resolution 3D-content utilizing all four layers, the frame rate is constantly above 60Hz. Content can be provided by a PC and esp. for the FPGA-solution by a set-top box, a gaming console or the like – which is like driving a normal 2D- or 3D-display.

Both solutions have a very scalable design to already incorporate capabilities for more and larger Sub-Holograms, especially for Sub-Holograms spreading in both dimensions (full-parallax holography). Even with the current solutions

the calculation of full-parallax color holograms in real-time is achievable and has been tested internally. Dependent from 3D-scene complexity there may be some restrictions to smaller depth-ranges according to the available computing power. Already today, by using multiple GPUs (NVIDIA SLI or AMD CrossFire) or linking multiple FPGAs, the calculation-performance can be sufficiently increased easily. As an example, NVIDIAs SLI has been applied to connect two GeForce 285 GTX GPUs which increased the hologram-calculation frame rate by a Factor of 1.9, proving the good scalability of this solution.

These solutions enable SeeReal to show complex 3D-videos as well as complex interactive 3D-scenes, which are all holographically encoded on-the-fly. This now allows focusing more on the development of holographic content and applications along with appropriate formats for streaming, gaming or holographic TV, than on the technology itself.

## 10. CONCLUSION AND FURTHER DEVELOPMENT

This paper presented our solution using the novel Sub-Hologram approach, which allows the calculation of holograms in real-time when adapted to GPUs or FPGA-hardware for driving holographic displays showing complex and interactive 3D-content.

Further developments will concentrate on new technologies in the area of GPUs like Direct3D 11 with the newly introduced Compute-Shaders and OpenGL 3/4 in combination with OpenCL, to improve the efficiency and flexibility of SeeReal's GPU-solution. SeeReal's FPGA-solution will be completed in 2010 including the support for multiple transparency-layers. Furthermore the VHDL-designs will be optimized for the development of dedicated holographic ASICs.

The development or adaption of suitable formats for streaming holographic video (3D-TV) and 3D-content along with the integration of SeeReal's technology into existing game- or application-engines will be another focus.

## REFERENCES

- [1] Goodman, J.W., [Introduction to Fourier Optics], 2nd edn, McGraw-Hill, New York (1996).
- [2] Hoffman, D. M., Girshick, A. R., Akeley, K. & Banks, M. S., "Vergence-accommodation conflicts hinder visual performance and cause visual fatigue," *J. Vis.* 8(3), 1–30 (2008).  
<http://journalofvision.org/8/3/33/>
- [3] St-Hilaire, P., Benton, S. A., Lucente, M. E., Sutter, J. D. & Plesniak, W. J., "Advances in holographic video," *Proc. SPIE* 1914, pp. 188–196 (1993).  
<http://link.aip.org/link/?PSI/1914/188/1>
- [4] Burckhardt, C. B., "A Simplification of Lee's Method of Generating Holograms by Computer," *Appl. Opt.* 9(8), 1949–1949 (1970).  
<http://ao.osa.org/abstract.cfm?URI=ao-9-8-1949>
- [5] Hsueh, C. K. & Sawchuk, A. A., "Computer-generated double-phase holograms," *Appl. Opt.* 17(24), 3874–3883 (1978).  
<http://ao.osa.org/abstract.cfm?URI=ao-17-24-3874>
- [6] Gregory, D. A., Kirsch, J. C. & Tam, E. C., "Full complex modulation using liquidcrystal televisions," *Appl. Opt.* 31(2), 163–165 (1992).  
<http://ao.osa.org/abstract.cfm?URI=ao-31-2-163>
- [7] N. Leister, A. Schwerdtner, G. Fütterer, S. Buschbeck, J.-C. Olaya and S. Flon, "Full-color interactive holographic projection system for large 3D scene reconstruction," *Proc. SPIE* 6911, 69110V (2008).  
<http://dx.doi.org/10.1117/12.761713>
- [8] Häussler, R., Reichelt, S., Leister, N., Zschau, E., Missbach, R. & Schwerdtner, A., "Large real-time holographic displays: from prototypes to a consumer product," *Proc. SPIE* 7237, p. 72370S (2009).  
<http://link.aip.org/link/?PSI/7237/72370S/1>
- [9] <http://www.intel.com/support/processors/sb/cs-023143.htm>
- [10] <http://www.business-sites.philips.com/3dsolutions/home/index.page>
- [11] <http://en.wikipedia.org/wiki/Shader>
- [12] [http://www.nvidia.com/object/cuda\\_home\\_new.html](http://www.nvidia.com/object/cuda_home_new.html)
- [13] <http://gpgpu.org/about>