DOI: 10.1111/cgf.70058
EUROGRAPHICS 2025 / A. Bousseau and A. Dai
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 44 (2025), Number 2

Real-Time Rendering Framework for Holography

Sascha Fricke[1,2], Susana Castillo[1,2], Martin Eisemann[1], and Marcus Magnor[1,2,3]

[1]Computer Graphics Lab, TU Braunschweig, Germany *{lastName}@cg.cs.tu-bs.de*
[2]Cluster of Excellence PhoenixD, Leibniz University Hannover, Germany
[3]Dpt. Physics and Astronomy, University of New Mexico, USA

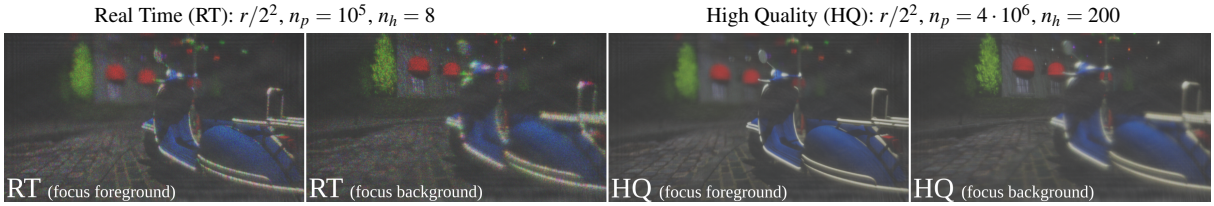

Figure 1: ***We introduce a fast rendering framework for Point-Based Computer Generated Holography.*** *Our system is capable of generating visually pleasing holograms in real time (left), while adaptively changing the amount of points for better quality results (right), providing a flexible way of finding the right balance. Furthermore, our method accounts for visibility changes due to parallax and depth of field.*

Abstract

With the advent of holographic near-eye displays, the need for rendering algorithms that output holograms instead of color images emerged. These holograms usually encode phase maps that alter the phase of coherent light sources such that images result from diffraction effects. While common approaches rely on translating the output of traditional rendering systems to holograms in a post processing step, we instead developed a rendering system that can directly output a phase map to a Spatial Light Modulator (SLM).

Our hardware-ray-traced sparse point distribution, and depth mapping enable rapid hologram generation, allowing for high-quality time-multiplexed holography for real-time content. Additionally, our system is compatible with foveated rendering which enables further performance optimizations.

CCS Concepts

• ***Computing methodologies*** → ***Rendering;*** *Virtual reality;* • ***Hardware*** → ***Emerging technologies;***

1. Introduction

The current generation of Virtual Reality (VR) devices is typically constructed using amplitude modulating Liquid Crystal Displays (LCDs) or Organic Light Emitting Diodes (OLEDs). The primary disadvantage of these devices is their bulkiness and weight, which can impede comfort during extended use. Additionally, they lack the information necessary for correct depth perception, as they display a sequence of 2D images at a fixed distance, which can strain the eyes over time. Furthermore, the process of shrinking these devices rapidly approaches the diffraction limit. This is a constraint that limits the minimum size and, consequently, the weight that can be achieved in the construction of these devices. As a result of the amplitude-modulating functionality inherent to these devices, the image brightness and contrast become a limiting factor, particularly in smaller devices.

Recently, there has been promising research exploring new display technologies that utilize devices that do not rely on amplitude modulation, but rather on phase modulation [KGC*22]. By altering the phase of a coherent light source, and keeping the amplitude constant, an image can be formed through interference effects caused by the wave properties of light. This mechanism is commonly referred to as holography. With this approach, a light field can be reproduced instead of a single 2D plane thereby enabling correct depth of field perception. This allows the user to exercise full focal control over the observed content. Furthermore, to a certain extent, it even allows parallax, which has recently been demonstrated to be a significant element in fostering immersion [KNC*24].

Previous methodologies have primarily concentrated on the development of algorithms that facilitate the conversion of existing content, created with conventional 2D devices in mind, into phase modulation maps for output on Spatial Light Modulator (SLM) devices. This information is typically augmented with depth data, which is either extracted from the synthesis process or captured using RGB-D cameras to generate 3D holograms. While these methods

are capable of producing high-quality holograms, they all assume that the content is already created. An alternative approach is to generate holograms directly from 3D information using more flexible and sophisticated methods [BCKS21, MBS21]. This allows the capture of the entire scene, including occluded areas. However, these methods are still too expensive for real-time applications and are only used to train machine learning models that are then used to synthesize holograms [SLK*21].

In this work, we present a rendering algorithm that replaces the classic rendering pipeline found in current generation rendering systems – which are designed with traditional display technology in mind – and directly output a phase map to a SLM, circumventing the conventional intermediate step of image synthesis and conversion.

Our approach builds upon the real-time Point-Based method initially proposed by Fricke *et al.* [FCC*24] to distribute points on the surface of triangular objects using hardware ray-triangle occlusion tests to occlude parts of the spherical wavefront that is emitted by individual points. This approach has been demonstrated to be effective, as it is scalable to arbitrary precision and closely aligns with the fundamental Huygens principle of wave optics. However, a limitation of this method is that, although it is trivially scalable in terms of quality, it has previously been cost-prohibitive for real-time use due to its suboptimal performance scaling characteristic, as its computation time is proportional to the number of points. We extend Fricke *et al.* by several key components that make it suitable for VR applications. Our method supports color holograms by generating three holograms simultaneously and adaptively distributing the fixed point budget between them. We furthermore developed a highly scalable hardware-ray-tracing based point scattering method that correctly accounts for visibility changes due to parallactic viewpoint changes and Depth of Field (DoF). Finally we remapped the depth range of the resulting point based hologram such that the performance is significantly improved while not altering the perspective projection in any way. Nevertheless, our findings indicate that, akin to the conventional triangle rasterization approach, this method can be accelerated to a level that makes it viable for real-time usage. The number of points required for good quality can be maintained at a constant level, irrespective of the scene configuration, provided that the points are distributed in an adequate manner. One of the principal advantages of Fricke and colleagues' approach is its intrinsic sparsity, which renders it particularly well suited for augmented and foveated rendering by merely adjusting the point distribution to attain the desired target point density in the fovea, as proposed by Hong *et al.* [HKH*16]. As a consequence of the superposition principle, our method also conveniently scales to arbitrary precision when progressively computing points in a sequential manner. By varying the point density and the overall number of points, our method provides a flexible means of identifying the optimal balance between quality and performance for the intended application. This is illustrated in Fig. 1.

Our main technical contributions can be summarized as follows:

- A fast and flexible point scattering method that accounts for parallax and DoF effects that can be tuned to specific device requirements.
- An extension of a real-time Point-Based Holography (PBH) algorithm to allow creation of multiple holograms simultaneously with adaptive point distribution between different color channels.
- A non-linear depth remapping to overcome the limitations of common PBH methods to allow arbitrary scene configurations.

2. Related Work

The concept of holography was first invented by Dennis Gabor [Gab48], who envisioned a new imaging method that would capture the phase distributions of phase-altering inhomogeneous participating objects. Through phase unwrapping these could then be converted to depth images. Today, the field of holography can be roughly divided into two subcategories: on the one hand Digital Holography, which deals with imaging algorithms similar to what Gabor first proposed; and, on the other hand, Computer Generated Holography (CGH), which tackles the inverse problem. In this latter category, instead of capturing interference patterns, similar patterns are generated to reconstruct the original light fields as if emitted directly from virtual objects. Most often, those are phase only distributions due to limitations of current display technology. In this section, we focus on CGH algorithms to give an overview of the core concepts that have been established in scientific literature as well as a brief explanation of their strengths and weaknesses.

Computer Generated Holography has been a long established field, and dates back as far as 1966 when it was invented by Brown and Lohmann [BL66]. Decades later, Petz *et al.* [PM03] utilized graphics hardware to accelerate holography trying to bring it closer to real time. Since then, many approaches have been found to both accelerate this process, and improve quality. For a comprehensive overview of the topic, we refer the reader to the survey paper by Sahin *et al.* [SSMG20]. Overall, the whole field can be roughly subdivided into three general categories: Layer-Based, Holographic Stereograms, and Point-Based. We will further discuss them in detail in the following sections. Additionally and in recent times, a fourth category has emerged that exploits the advantages of machine learning [PCPW20, CGP*21, SLK*21, LKL*22, CGP*22, LWHC23, GLC*24]. This category draws on the preceding three categories but also incorporates novel ideas. Its use in CGH has gained traction in recent years due to the promise of constant time complexity independent of scene complexity. Despite the absence of machine learning in our methodology, we believe it merits mention due to its significant surge in popularity. For a detailed look at this topic we recommend the review by Shimobaba *et al.* [SBB*22].

2.1. Layer-Based

The Layer-Based method was first introduced by Bayraktar *et al.* [BO10] and consists of subdividing the scene along the view axis into multiple layers using for example a depth image, which is either derived from the image synthesis process (*e.g.*, [CGP*21]) or captured using RGB-D cameras (*e.g.*, [SLK*21]). These layers can then individually be propagated towards the hologram plane using either iterative plane-to-plane propagation models (see [Lat19]) or through learned models (*e.g.*, [CGP*21]).

These methods are popular due to constant computation time independent of the scene configuration, propagation distance, pixel pitch, or wavelength of the light source. The Layer-Based method only

scales with target resolution and number of layers. However, plane-to-plane propagation models are prone to aliasing artefacts with growing propagation distance, which can be alleviated by performing zero-padding when applying the Fast Fourier Transform (FFT), which in turn increases the computation time. Alternatively, the number of layers can be increased, which again negatively impacts the performance. Another problem is that with too few layers, artefacts can appear, especially when viewed off-axis due to the limited information captured in individual layers.

2.2. Holographic Stereograms

Another popular approach to CGH is to compute the hologram from a light field. The term *Holographics Stereogram* was coined by McCrickered and George in 1968 [MG68]. The concept of using graphical rendering in holography stems from early work on these stereograms, with one of the pioneers being Yamaguchi *et al.* [YHHO93], and later improvements proposed by Kang *et al.* [KYY08]. In CGH algorithms pertaining to this category, the hologram is not directly computed from 3D geometry; instead, a light field is captured and converted to a hologram. Rather than simulating the wave propagation process that generates the hologram, the angular spectrum that represents the equivalent visual result is approximated. The primary advantages of this approach are twofold. Firstly, the hologram computation is independent of geometric complexity. Secondly, occlusion is implicitly encoded in the light field. In many cases, the hologram generation process is divided into a grid of sub-holograms, which are referred to as *Hogels* [PPW19]. These Hogels encode individual view directions. In a recent study, Kim *et al.* [KNC*24] employed this approach to generate time-multiplexed high-quality holograms. However, this method is constrained by its limited angular resolution, which necessitates the use of numerous viewports to achieve sufficient resolution, making it prohibitively expensive for real-time applications.

2.3. Point-Based

The Point-Based method is one of the oldest methods, since it follows directly from the Huygens-principle. It was first proposed in 1966 by Waters *et al.* [Wat66]. This class of algorithms has been improved over the years in several ways from caching the expensive wavefront calculations in LUT [Luc93], reducing the size of those LUT, and even transforming the Point Spread Function (PSF) to and accumulating them in wavelet space [SMT*18]. All of these improvements were designed to improve the performance of hologram computation specifically on the CPU and, thus, were not suitable for real-time computation on consumer-grade hardware.

Petz *et al.* [PM03] were the first to utilize the graphics hardware to accelerate accumulation of the holograms to bring it closer to real-time. Although occlusion computation was tricky with this class of algorithms, Chen *et al.* [CW09] proposed an analytical method for this, which involved sorting the points first to be processed in a front-to-back order.

One established method to limit the computation time due to the propagation distance is the use of what is known as the Wavefront Recording Plane (WRP), which was introduced by Shimobaba *et al.* [SMI09]. To further improve the performance, using multiple wavefront recording planes have been proposed ([SBMS15]). A careful balance however has to be chosen between point propagation distance and number of wavefront recording planes to gain any advantage, which can be highly scene dependent.

The Point-Based method offers the most flexibility since it works with arbitrary scene configurations and depth distributions. The biggest limitation of the Point-Based method however is, that its performance is not only impacted by the number of points, but also by the propagation distance and the wavelength. Most recently, Fricke *et al.* [FCC*24] utilized the graphics hardware to accelerate this method in a novel way using the rasterization hardware, which avoids complex thread scheduling and enables efficient occlusion culling. We build upon this approach because of its aforementioned flexibility albeit being too expensive for real-time context. These recent advancements might have helped lift this limitation or, at least, bring this method closer to real-time.

3. Method

We build upon the approach of Fricke *et al.* [FCC*24], and extend it to suit Extended Reality (XR) applications. They focused on a novel approach to efficient GPU-driven point-based computer-generated holography using the mesh shading pipeline, which provides a promising foundation for XR. First of all, we provide a method of utilizing the graphics hardware to render several holograms simultaneously that takes advantage of the sparsity of the Point-Based (PB) method, enabling color holograms with minimal overhead. Furthermore, their point scattering approaches did not scale well to arbitrary scene configurations. Their mesh based scattering method was completely static, which fully eliminates the overhead of placing points at runtime but this made it also very sensitive to scene complexity, which renders it unsuitable for arbitrary virtual environments. Their alternative approach of scattering points based on a grid of G-Buffers is more flexible but again sensitive to G-Buffer resolution and number of viewports. We instead developed a hardware-ray-tracing based scattering method, that scales very well to arbitrary scenes and point budgets and does not waste computational effort. Finally, one of the major limitations of the PB method is, that the performance decreases with the distance of the points to the plane, where the wavefront is recorded. We however observed, that the necessity for placing the points at the correct depth decreases with distance and remap the depths in such a way, that there is a predictable computational upper bound, that does not interfere with the perspective in any perceivable way.

The fundamental idea involves approximating the 3D scene as a collection of points. According to the Huygens principle, these points emit spherical electromagnetic waves (represented by complex values) to the destination plane. The accumulation of values from all points results in the target wavefront. The phase of this resulting wavefront in the target plane is what we refer to as the hologram. The whole system can be subdivided into point scattering, irradiance computation, point projection, rasterization, wavefront recording plane computation, and, finally, post-processing, and phase conversion. Details about these individual steps are described in the following sections, and a graphical overview of the method is given in Fig. 2.

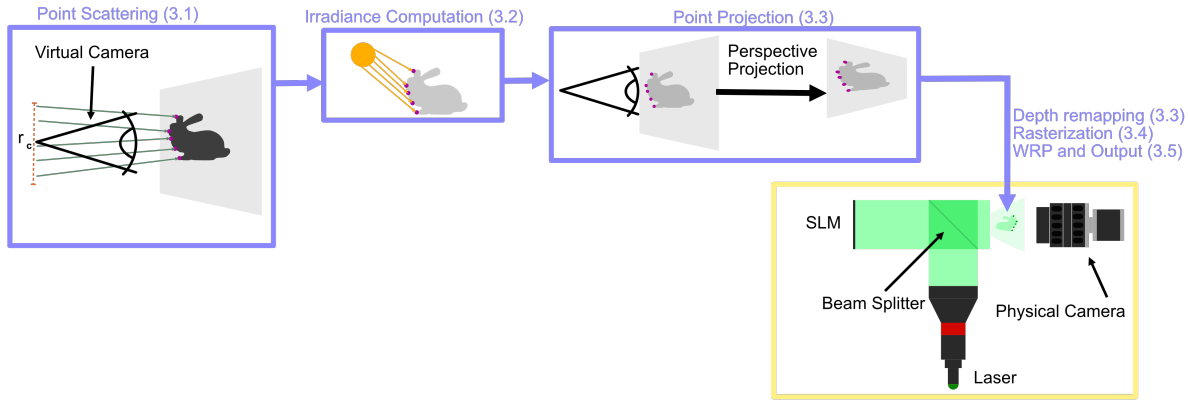

Figure 2: ***Overview of our method.*** *First, points are scattered by casting rays into the scene from the camera's point of view. The projection center is randomly selected within a user-defined circle of confusion to capture the full visible area corresponding to the spatial and angular potential of the SLM. We then gather diffuse irradiance on those point positions. After this, the points are projected to NDC space. The scene depth is remapped independently to the desired output range for hologram computation. Finally, the hologram is rasterized, propagated to the target distance, phase components are extracted and output to the SLM. The purple framing represent parts of the pipeline corresponding to the CGH algorithm, while the yellow frame corresponds to the physical setup at the end illustrating the projected target geometry.*

3.1. Point Scattering

Since the point-based holography method relies on a point cloud as the input geometry representation, we need to choose an efficient method of extracting points from the usual triangle based scene representation. For this, Fricke *et al.* [FCC*24] proposed several approaches. Scattering points directly on the triangular surface has the advantage that no scattering operations need to be performed during the generation of the hologram. Similar to the vertex processing step of classical rendering pipelines, where triangular vertex points are transformed to view space, the same can be done for object points, making the point representation fairly convenient for real-time rendering. However, for more complex VR applications, the total point budget quickly becomes a bottleneck. The alternative approach discussed in the paper by Fricke *et al.* [FCC*24] is, however, still too expensive for real-time applications because of the necessity of rasterizing the scene from multiple viewpoints, computing the shading densely on those viewports, and the Cumulative Distribution Function (CDF) computation and binary search operations. Even without the importance sampling step proposed by Fricke *et al.*, this method is prohibitively expensive and wasteful, since due to the sparse subsampling the authors proposed, vastly more samples are generated than necessary. Furthermore, if the gathered samples were fully utilized, the sampling would be limited to regular grid sampling, which can introduce undesired artefacts.

Rasterizing only a single G-Buffer from the point of view of the camera and then sub-sampling that to generate the point information is a cheaper alternative but it introduces visibility limitations, which become apparent in the presence of DoF and when viewed off-axis. One of the benefits of holography though, is that a hologram can encode a bigger part of the light field than a traditional image can. Although near-eye displays do not enable the user to perceive parallax effects directly, since the display is fixed in front of the eyes, recent studies have shown that this effect is still important for immersion (see [KNC*24]). We hypothesize that this is due to the occluded areas becoming visible due to DoF. We therefore choose a point scattering method that is flexible enough for full VR environments, while being scalable to achieve real-time framerates on a variety of devices. Our method closely resembles the common numerical method for computing DoF. Rays are traced into the scene starting at a point on a circle of confusion around the projection center. These rays are traced through the center of the pixel. Bottom left of Fig. 2 illustrates this method. In principle, the radius of the circle of confusion (r_c) can be freely chosen as long as it is large enough to capture all potentially visible surfaces. Occlusion computation is handled separately and guarantees a correct result in the end. However, due to the limited point budget, we would like to make sure that we get a coverage that is as close to ideal as possible. The radius depends on the pixel pitch of the SLM since that defines the maximum viewing angle of the hologram, and thus the amount of parallax we have to account for. The relation is given by the grating equation:

$$\sin\theta_{\max} = \frac{\lambda}{2p}. \tag{1}$$

The maximum viewing angle of the hologram is $\theta_{\max}$ and p is the pixel pitch of the SLM. The correct viewing angle can therefore be found by computing the size of the circle of confusion such that the maximum divergent angle is equal to $\theta_{\max}$. The radius of confusion can then be calculated as: $r_c = \tan\theta_{\max}$. This method is reminiscent of Zhang *et al.* [ZCC*11]. However, they sampled the wavefield directly using raytracing, which is significantly more expensive. We, however, can perform this ray-casting process vastly more sparsely since the wave propagation is done later in a separate step.

Note that even though this point distribution is sparse even for VR applications, the image formation process works differently from regular image synthesis. Visual noise is not necessarily perceived due to the sparsity of points, since due to the diffraction effects and the distribution of points in 3D, no point will be perfectly in focus

and they will visually merge together. Visual noise is mainly dictated by what is known as speckle noise, which results from destructive interference of neighboring points. A visualization of these effects is provided in our supplemental material.

3.2. Irradiance Computation

After the points are scattered, lighting computations are performed. We store irradiance values that are gathered using hardware ray-tracing on the points. Irradiance is computed using traditional RGB color values. Due to the fact that holography depends on the use of coherent light sources, RGB holograms can only be achieved by computing three separate holograms and either combining them with three separate SLM devices or by multiplexing them temporally. We achieve this by computing three holograms at the same time and distributing our given point budget to the three separate color channels. The point channel to which a given point belongs is chosen stochastically based on the captured irradiance value. Given a total point budget of n_p points, we assign a point to a specific color channel c_k in the following manner: We first derive a color selection probability for the i^{th} point from the diffuse irradiance value $\vec{c}$ gathered at the point as:

$$p_i = \frac{\vec{c}}{c_0 + c_1 + c_2}, \tag{2}$$

we can then choose the point channel k efficiently as:

$$k = \begin{cases} 0 & \text{if } \xi_i < p_{i,x} \\ 1 & \text{if } \xi_i < p_{i,x} + p_{i,y} \\ 2 & \text{otherwise} \end{cases} \tag{3}$$

where ξ_i is a uniform random variable that is unique for every point. In our tests, deriving a random value by hashing the point index was sufficient.

3.3. Point Projection

The points are projected to NDC space, similar to what we would do in a vertex shader. That way, we are flexible with regard to the perspective, as the Field of View (FOV) would otherwise be dictated by the pixel pitch of the SLM, governed by Eq. (1).

The perceived scene scale can be found by dividing the width of the SLM (w_{slm}) by the image plane size. Since we assume a scene scale of 1 scene unit equal to 1 meter, we can compute this as $s = 0.5\, w_{\text{slm}}$. However, this results in a very small scene scale and a strong DoF effect. We can achieve effective scene scaling by remapping the depth range. This also has a positive impact on performance since the computational cost of Point-Based holography is known to scale proportionally with the propagation distance. The depth value is taken in clip space, before perspective divide, normalized, and then remapped to the desired range, and then passed to the later stages of the pipeline. The remapping is done according to:

$$d' = d_{min} + (d_{max} - d_{min}) \cdot \mathrm{S}\left(\frac{d - d_{near}}{d_{far} - d_{near}}\right). \tag{4}$$

Here, d is the original clip space depth value, d' is the remapped value that is used later for hologram computation, and $\mathrm{S}(t)$ is the normalized smoothstep function, which is given by $\mathrm{S}(t) = t^2 \cdot (3 - 2t)$, guaranteeing a smooth transition to a constant max depth. d_{min} and d_{max} specify the depth range that is used for hologram calculation, and d_{near} and d_{far} specify the scene space depth ranges that will be remapped to the desired output depth range.

With these formulas we can compute the radius $r_{d'}$ of a disk that projects the spherical phase and amplitude distribution for a single point at distance d' onto the WRP by evaluating:

$$r_{d'} = r \cdot d' \cdot \tan \theta_{\max}, \tag{5}$$

where d' is the remapped distance of the point to the WRP. As computation time scales with disk radii, we introduce a parameter r as a scaling factor to control the overall performance. Notice that this formula is remarkably similar to the one defining r_c because both depend on the same angle. The third step on Figure 2 illustrates the resulting geometry that is encoded in the hologram. Notice that this remapping of the depth range does not impact the perspective in any way. The perspective divide is still done with the original values according to the traditional perspective projection matrix provided. The new depth values are only used for the computation of the projected disk radius of the PSF in Eq. (5) and the phase and amplitude computation, which will be described in the following section.

3.4. Rasterization

Here we use the mesh shading pipeline to compute the coverage of the point spread functions based on the method of Fricke *et al.* [FCC*24], who approximated the radially symmetric point spread functions of individual points using procedurally generated triangle fans. This is a useful approximation since, due to occlusions and shading effects, the projected area of a point spread function can become arbitrarily complex. Using hardware rasterization to perform this task simplifies scheduling and allows it to be fully GPU-driven. Hardware ray-traced occlusions are used to discard individual segments. The original authors did not mention the exact parameters they used for their implementation. However, they mentioned that they spawned one Task Shader workgroup per object point, 16 threads per Task Shader workgroup, and one Meshlet (Mesh Shader workgroup) per Task Shader Thread. We instead choose to compute a whole disk in one Mesh Shader workgroup to increase GPU utilization. We try to avoid too many triangles since rasterization is known to perform better with larger triangles. However, too few triangles limit the granularity with which we can discard individual disk segments. Thus, we choose 16 triangles per disk and one occlusion ray per triangle. To avoid thread divergences we output a fixed amount of triangles from the Mesh Shader but degenerate the triangles by setting the indices of a triangle to its first vertex point. In our tests, we observed similar or better performance benefits by doing this compared to varying the number of triangles and computing correct offsets, which also significantly increases the complexity of the computation. Computing the occlusion by spawning occlusion rays from the center of the triangles limits the occlusion approximations to a fixed angle $\theta_{\max}$, which becomes especially apparent for smaller pixel pitches. Thus, we chose to uniformly randomly vary the starting point of the occlusion rays within each individual triangle. In this way, given an adequate amount of points overall, we get a better approximation of the correct visibility.

We use multi-view rendering to distribute the points to individual color channels. Based on the previously assigned color channel according to Eq. (3), one of the three viewports is selected, which allows efficient scheduling and generation of three separate holograms simultaneously. Since the image formation process in holography relies on coherent laser light, the resulting image is subject to speckle noise resulting from unwanted interferences. A known method to prevent this is to separate the light in the temporal domain by generating multiple holograms in quick succession [CGP*22]. This process is known as *time-multiplexing*. We can make use of our multi-view hologram computation for this process as well, which can be seen in the supplemental material.

Each point k spherically projects a complex wavefront onto a plane at a distance d'. The complex amplitude $U_{i,d'}$ is computed as:

$$U_{i,d'}(x,y) = U_i e^{\frac{j2\pi z_{x,y}}{z_{x,y}\lambda}}. \tag{6}$$

Here, j is the imaginary unit, $z_{x,y}$ is the distance from the i^{th} point to the destination pixel on the target plane. While it is a common practice to precompute this in a Look Up Table (LUT), we did not find a significant speed improvement when precalculating PSF values and looking them up from a LUT during rasterization. We instead choose a fast sine approximation:

$$\sin x \approx 4\left(\left(\left(\frac{x}{\pi}+1\right)\%\,2\right)-1\right)\left(1-\left|\left(\left(\frac{x}{\pi}+1\right)\%\,2\right)-1\right|\right),$$

with its cosine counterpart:

$$\cos x \approx 4\left(\left(\left(\frac{x}{\pi}-\frac{1}{2}\right)\%\,2\right)-1\right)\left(1-\left|\left(\left(\frac{x}{\pi}-\frac{1}{2}\right)\%\,2\right)-1\right|\right).$$

Here, % denotes the modulo operation, which computes the remainder of the integer division operation. Using Euler's formula we can efficiently compute $U_{i,d'}$ for each point. Each point contribution is at the end accumulated on the target plane by the hardware rasterizer by evaluating Eq. (6) for each point in the fragment shader.

3.5. Wavefront Recording Plane (WRP) and Output

Since, as mentioned before, computational effort grows with propagation distance in case of the Point-Based holography method, the wavefront is usually captured in a plane as close to the scene content as possible (see [SMI09]). From here, a plane-to-plane propagation method is used, in our case the angular spectrum method. The advantage of this method is that the FFT can be used to compute this propagation, which computationally scales much more predictably and favorably than the Point-Based method. At the end, we derive the phase from the complex wavefront and output it to the SLM.

4. Results and Discussion

All results were created using a NVidia GeForce RTX 4090. The SLM used is a HoloEye Luna VIS which has a pixel pitch of 4.5 µm and a resolution of 1920×1080. The device is capable of a refresh rate of 60 Hz or 180 Hz in Color Field Sequential (CFS) mode. The illumination source is a Thor Labs 520 nm laser. The whole setup is shown in Fig. 3. A beam splitter is used to redirect the laser onto the SLM, which reflects the phase modulated light through a second beam splitter, that redirects it to a magnifying glass for direct observation. The remaining light is directed towards a Basler a2A2590-60umBAS camera, where it is captured for evaluation. Our camera setup operates in linear space. All images are captured, combined and accumulated in linear space. Gamma correction is applied at the end. The conversion from complex wavefront to phase only wavefront is done by simply discarding the amplitude. Double Phase Amplitude Coding (DPAC) as presented by Hsueh and Sawchuk [HS78] is not applicable in our particular case since non-normalized diffuse irradiance is stored on the points and used to weight the points. A global normalization would necessitate a reduction over all points, which is, given the very tight time budget, too expensive. A simple local tone-mapping, by applying $1/(1+x)$ before performing DPAC, is not sufficient due to its reliance on forcing incoherence to modulate the brightness (*i.e.*, it lowers brightness everywhere except in areas where the amplitude is one, which is never the case with local tone-mapping).

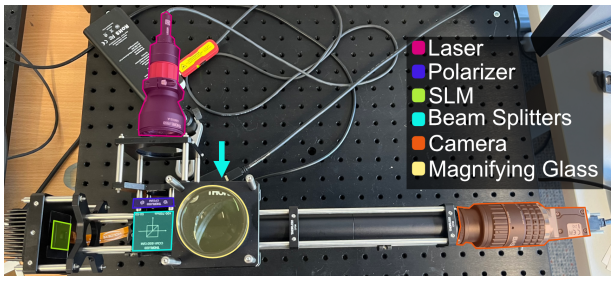

Figure 3: ***Capturing Setup.*** *The laser is redirected onto the SLM using a beam splitter. The SLM applies the phase shift as controlled by our system. The reflected wavefront passes through a second beam splitter (positioned under the magnifying glass), where it is redirected to a magnifying glass for direct observation. The remaining wavefront travels straight into a camera for image capturing.*

For all experiments, we vary the number of points (n_p) between 10^4 and 4 million points. We found that $n_p \in [10^5, 10^6]$ is the range which gives the best trade-off between quality and computational time, which is why we introduced a further split in that range (*i.e.*, $n_p = 5 \cdot 10^5$). All time values are for computations of three holograms for the red, green, and blue color channels combined. The depth range is kept to a maximum of 9 mm. This might seem small, but we found it to be a reasonable trade-off between the amount of depth of field and computational time.

In summary, our experimental parameters are as follows: scaling factors for the radii $r \in \{1, 1/2, 1/2^2, 1/2^3, 1/2^4\}$, number of points $n_p \in \{10^4, 10^5, 5 \cdot 10^5, 10^6, 4 \cdot 10^6\}$, and number of time-multiplexed holograms $n_h \in \{1, 3, 8, 20, 100\}$. A visual comparison for depth range and performance impact is given later in this section. The maximum scene scale depth that is used for the mapping explained in Eq. (4) is set to 10 units. In the following sections, we evaluate how the the various parameters of our presented method influence the performance and image quality.

4.1. Time Multiplexing

Choi *et al.* [CGP*22] has shown that time multiplexing holograms can increase quality in the context of heavily quantized phase out-

puts. Due to the sparse nature of our algorithm, we want to examine if time-multiplexing highly sparse holograms can have a similar benefit. Thus, we captured multiple holograms with both varying total numbers of points and varying amounts of holograms in total. Since we did not have access to a high frequency SLM as of the writing of this manuscript, we averaged the holograms together after capturing. The numerical evaluation of the results, according to quality metrics PSNR, SSIM [WBSS04], LPIPS [ZIE*18], are presented in Tab. 1. The corresponding visualizations are given in our supplemental. According to Tab. 1 we can see that it is still beneficial to distribute points temporally to avoid coherent interferences between neighboring points. For example computing $n_h = 3$ holograms with 10^5 points result in better LPIPS and SSIM scores than $n_h = 1$ hologram with 4×10^6 points but according to Tab. 2 the total computation time can be reduced from 45.25 ms to 33.57 ms. The visual counterpart of this example is provided in Fig. 4.

4.2. Color images

Our SLM is capable of outputting phase images at a frequency of 180 Hz addressed as RGB images at 60 Hz. However, we did not have access to a fiber-coupled RGB Laser for our experiments. Therefore, we chose to utilize the high frame rate for time multiplexing. Colors are produced in post production by outputting and capturing individual holograms sequentially. Notice that producing images with different wavelengths and thus different diffraction angles for individual color channels would lead to slightly different results. We do not expect these differences to be meaningful for the presented results though, but would like to have the opportunity to further examine this in the future. Figure 5 shows the impact of our color selection theme. Both images are captured with the same amount of points. We observe a significant overall noise reduction.

n_h	n_p	Metrics PSNR↑	SSIM↑	LPIPS↓
1	10^4	13.855	0.403	0.725
	10^5	14.583	0.375	0.734
	$5 \cdot 10^5$	14.750	0.374	0.731
	10^6	14.773	0.374	0.727
	$4 \cdot 10^6$	14.795	0.375	0.729
3	10^4	14.234	0.449	0.695
	10^5	14.791	0.433	0.684
	$5 \cdot 10^5$	15.024	0.442	0.670
	10^6	15.033	0.443	0.669
	$4 \cdot 10^6$	15.043	0.443	0.672
8	10^4	14.998	0.520	0.626
	10^5	15.274	0.562	0.570
	$5 \cdot 10^5$	15.321	0.575	0.547
	10^6	15.331	0.577	0.541
	$4 \cdot 10^6$	15.331	0.578	0.538
20	10^4	15.406	0.584	0.558
	10^5	15.470	0.650	0.469
	$5 \cdot 10^5$	15.481	0.664	0.448
	10^6	15.493	0.665	0.444
	$4 \cdot 10^6$	15.480	0.667	0.443
100	$4 \cdot 10^6$	15.491	0.732	0.367

Table 1: ***Quality metrics****. We report the impact on the final hologram's quality of different combinations of number of time multiplexed holograms (n_h) and number of points (n_p) used for each individual hologram.*

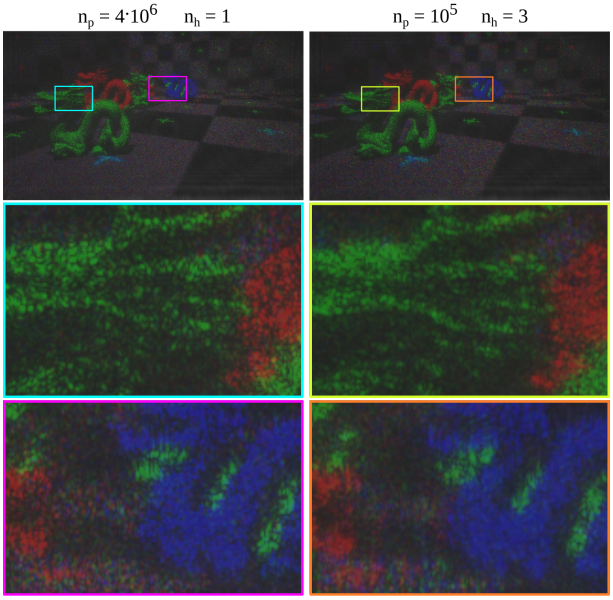

Figure 4: ***Time Multiplexing.*** *Computing more holograms with a lower amount of points each can not only increase the quality of the results but also have a beneficial impact on performance.*

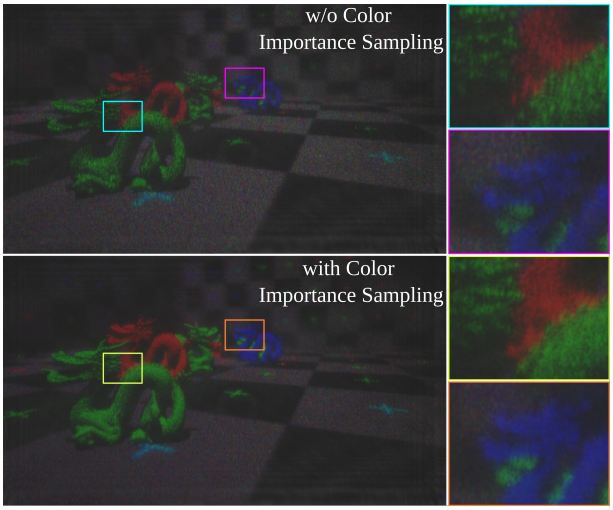

Figure 5: ***Color Selection.*** *Our importance sampled color channel selection scheme significantly increases the convergence rate and thus reduces the overall amount of points necessary. Both images are computed with $n_h = 8$ holograms of $n_p = 10^5$ points each.*

4.3. Point Scattering

As previously mention in Sec. 3.2, in order to generate color holograms, we have to create three separate phase images and distribute points across the three separate holograms. Thus, and instead of distributing the total amount of points to scatter evenly over the three color channels, we stochastically assign a point to a color channel based on the irradiance values. E.g., if the diffuse irradiance at a

n_p	Total Computation	WRP Computation	Avg. Time (SD) Point Scattering	Shading	Hologram Propagation
10^4	10.28 ms (1.34)	0.54 ms (0.05)	0.57 ms (0.48)	3.85 ms (0.82)	5.18 ms (0.38)
10^5	11.19 ms (0.74)	1.59 ms (0.05)	0.53 ms (0.07)	3.81 ms (0.70)	5.13 ms (0.06)
$5 \cdot 10^5$	16.29 ms (1.46)	6.24 ms (0.32)	0.74 ms (0.68)	3.94 ms (0.76)	5.23 ms (0.49)
10^6	22.66 ms (0.65)	12.07 ms (0.06)	0.90 ms (0.05)	4.32 ms (0.61)	5.20 ms (0.09)
$4 \cdot 10^6$	45.25 ms (0.62)	32.66 ms (0.06)	1.90 ms (0.31)	5.28 ms (0.37)	5.20 ms (0.44)

Table 2: ***Total and partial computational times*** *for hologram generation ($n_h = 1$) with different number of points, with a fixed radius r. Number in parentheses indicate the standard deviation.*

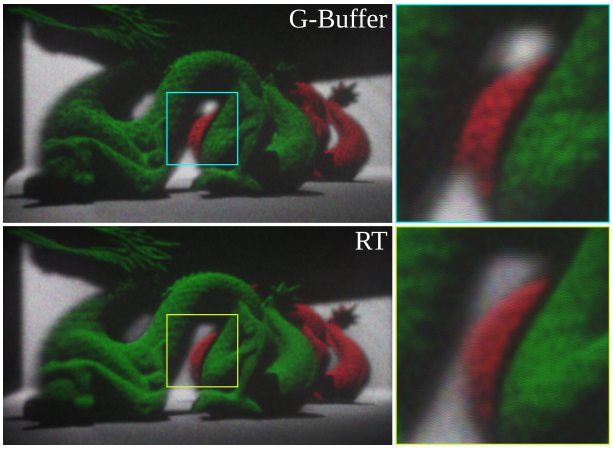

Figure 6: ***Point Scattering.*** *Difference of scattering points uniformly based on a rasterized G-Buffer vs. our ray-casting based approach. Placing points only on surfaces visible from a single point of view prevents the correct reconstruction of depth of field and parallax effects. Both images are simulated and reconstructed and averaged until convergence to isolate the point placement impact on the visibility.*

point is 100% red, the point would be assigned to the red channel with 100% probability. Table 2 shows the overall time of our method for different numbers of points. Point scattering time ranges between 0.57 ms and 1.9 ms.

Figure 6 illustrates the difference of scattering points from the point of view of the camera either with a naive G-Buffer based approach or our ray-casting method (see Sec. 3.1). Notice that the gap between the green dragon in the front and the red dragon in the back is not visible from the center of the camera. Our method is capable of correctly reconstructing the white wall in between whereas the G-Buffer method is not. Furthermore, areas that are out of focus appear with a dark contour due to the visibility of the wall behind not being correctly mixed in with the foreground. Our simulated reconstruction is based on the work of Ziegler *et al.* [ZBA*07].

4.4. Depth Remapping

We expect our depth remapping approach to have both a visual and a performance effect, as points that are further away than our defined maximum scene distance d_{far} are squeezed to a plane in the optical hologram reconstruction at distance d_{max}. Figure 7 illustrates this

Radius	Metrics PSNR↑	SSIM↑	LPIPS↓	Avg. Time
r	15.491	0.732	0.367	3.247 s
r/2	15.500	0.741	0.321	1.513 s
$r/2^2$	15.778	0.738	0.330	1.322 s
$r/2^3$	15.522	0.725	0.380	1.324 s
$r/2^4$	15.097	0.709	0.449	1.330 s

Table 3: ***Influence of radius on quality and computational time.*** *We report the impact of different radii on the hologram's quality according to different metrics. The results have been calculated with $n_h = 100$ and $n_p = 4 \cdot 10^6$.*

effect. Notice that the perspective still looks correct due to projecting all the points to NDC space before computing the hologram. The biggest effect of this remapping is the limited depth of field. This can be a useful and simple method of controlling the perceived scene scale. For bigger depth ranges, limiting the disk radius and thus cutting of higher angular frequencies might be a valid alternative or addition. We discuss this approach in Sec. 4.5.

4.5. Disk Radius

The main performance bottleneck in the Point-Based method in general results from the disk radius which scales with the distance of the point to the hologram plane (or WRP). This motivated our depth remapping approach. Another approach would be to simply limit the maximum disk radius and thus cutting off the angular frequencies recorded on the WRP directly. To examine this influence in detail, we reran the experiments with varying disk radii.

The visual impact of reducing the radius can be seen in Fig. 8. We report the computational time and quality metrics' results for $n_p = 4 \cdot 10^6$ and $n_h = 100$ in Tab. 3. A full split with all variations of number of points, is presented in our supplemental. We can see that reducing the radius is a simple but effective method of limiting the performance impact of large depth ranges. A curious observation is that the quality seems to increase for smaller radii as can be seen in Tab. 3. We hypothesize that this could be due to aliasing as the full radius disks contain angular frequencies close to the Nyquist limit.

4.6. Point Scattering Evaluation

Finally, we perform both qualitative and quantitative comparisons on our novel point scattering method against the method used by Fricke *et al.* [FCC*24]. To guarantee a fair comparison, we use the same lighting computations, depth remapping, and color channel assignment schemes as in our own work.

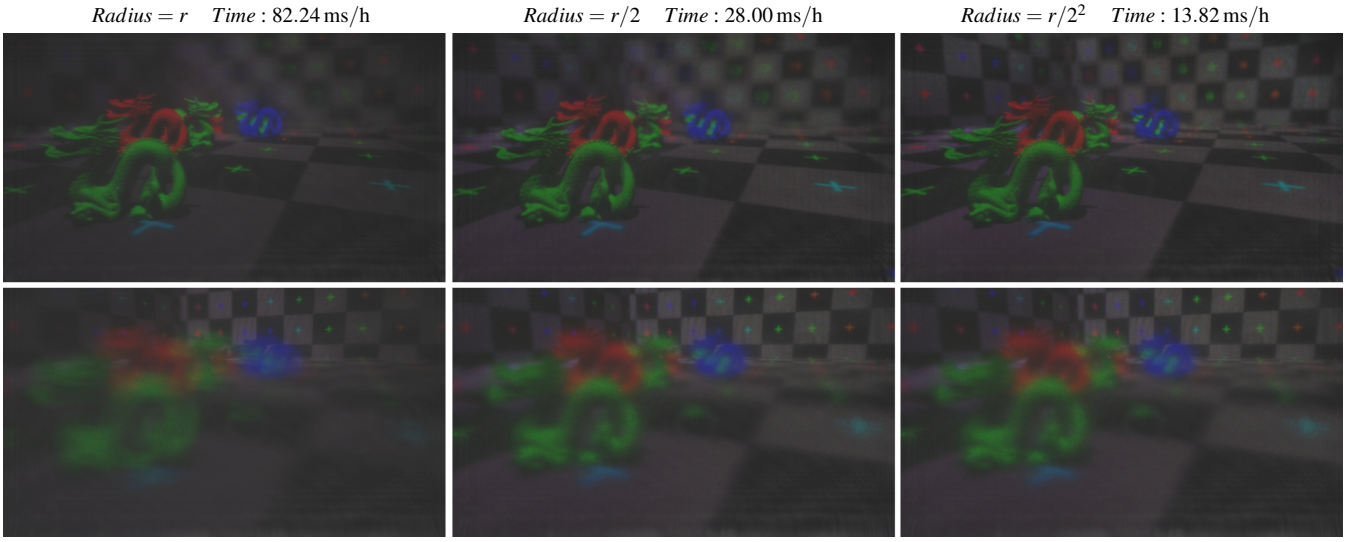

Figure 7: ***Influence of radii selection for scenes with large DoFs variation.*** *We show the visual and computational impact of altering the radius of the points (r) when computing scenes allowing for large DoFs. The top row has been calculated with focus on the foreground green dragon, while the bottom has the focus on the background wall. All results were generated with* $n_h = 200$*, and* $n_p = 4 \cdot 10^6$*.*

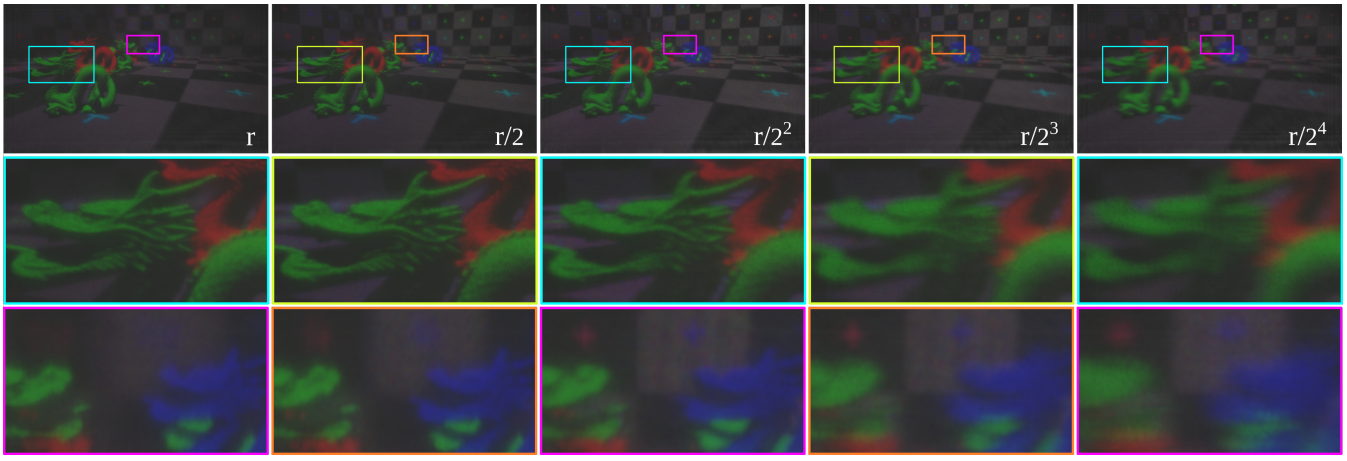

Figure 8: ***Influence of radius' size on image quality****. Optical hologram reconstructions with* $n_h = 100$ *holograms and* $n_p = 4 \cdot 10^6$ *points. Holograms are multiplexed close to convergence to better isolate and illustrate the impact of reducing the radius on the visual fidelity.*

Method	Viewports	Metrics PSNR↑	SSIM↑	LPIPS↓
Fricke *et al.* [FCC*24]	**2 × 2**	13.353	**0.630**	0.568
	3 × 3	13.463	0.628	0.573
	4 × 4	13.417	0.629	0.576
	5 × 5	13.511	0.627	0.581
Ours	-	**14.132**	0.627	**0.529**

Table 4: ***Quality comparison*** *against the work of Fricke* et al. *[FCC*24]. All comparisons are calculated for* $n_p = 4 \cdot 10^6$*,* $n_h = 8$*, and radius r.*

Table 4 shows the qualitative results obtained by the metrics, with the corresponding visual comparisons in Fig. 9. While both PSNR and LPIPS slightly favor our method, the viewport scattering method performs the best in terms of SSIM when using the fewest viewports. Nevertheless, our method clearly outperforms the viewport-based scattering in terms of computation speed as can be seen in the quantitative comparison displayed in Tab. 5. It becomes evident that the performance impact of using more viewports is significant compared to our method.

5. Limitations and Future Work

In contemporary rendering systems, temporal accumulation and reconstruction to increase sampling density have become integral components. Nevertheless, the application of simple Temporal Anti-Aliasing (TAA) with reprojection is ineffective, as the resulting

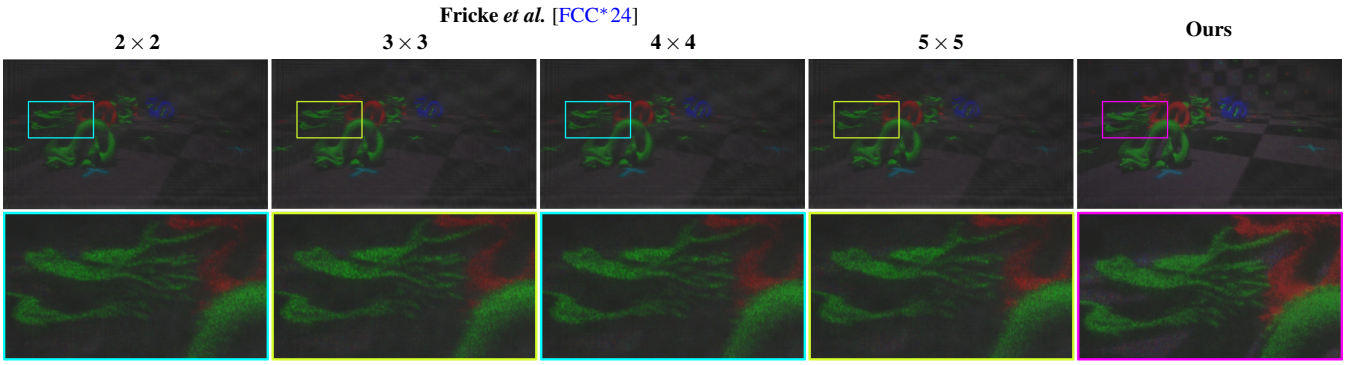

Figure 9: ***Visual comparisons*** *for different number of viewports for the method of Fricke* et al. *[FCC*24] and our approach. All results have been generated with* $n_p = 4 \cdot 10^6$, $n_h = 8$, *and radius r.*

n_p	Fricke *et al.* [FCC*24] Viewports 2 × 2	3 × 3	4 × 4	5 × 5	Ours
10^4	2.04 ms (0.10)	2.20 ms (0.10)	2.38 ms (0.11)	3.77 ms (0.08)	**0.57** ms (0.48)
10^5	2.26 ms (0.06)	2.25 ms (0.17)	2.59 ms (0.67)	3.83 ms (0.22)	**0.53** ms (0.07)
$5 \cdot 10^5$	2.43 ms (0.18)	2.51 ms (0.13)	2.84 ms (0.11)	4.19 ms (0.13)	**0.74** ms (0.68)
10^6	2.74 ms (0.03)	2.79 ms (0.07)	3.09 ms (0.26)	4.47 ms (0.10)	**0.90** ms (0.05)
$4 \cdot 10^6$	4.41 ms (0.10)	4.21 ms (0.14)	4.53 ms (0.10)	6.27 ms (0.39)	**1.90** ms (0.31)

Table 5: ***Computational time comparisons*** *against the work of Fricke* et al. *[FCC*24]. We compare the times to compute the point scattering of both methods when generating a single hologram with different number of points and a fixed radius r. All reported values are in milliseconds, with the values between brackets indicating the standard deviation.*

phase image can undergo unpredictable changes. This represents a limitation for all current holography systems. In static scenarios, the sampling density can be augmented; however, this is not possible in dynamic situations. Furthermore, in scenarios where a static background is present and a moving foreground is observed, it is possible to recompute only the portions of the hologram pertaining to the dynamic content, subsequently mixing them on top of the precomputed static content. In complex illumination setups, where radiance samples need to be temporally reused, a world space accumulation scheme analogous to that described by Binder *et al.* [BFK22] can be employed.

Our results indicate that the current primary source of overhead is the Angular Spectrum Method (ASM) -based propagation of the WRP to the hologram plane at the conclusion of the process. This is particularly evident in the context of time-multiplexing, where the resulting overhead is significant. One potential avenue for reducing this overhead is the utilization of performing fractional Fourier transforms, as previously demonstrated by Zahng *et al.* [ZCZ*17].

Our current system is configured to perform the computation of three holograms concurrently. The results presented in this paper demonstrate that rasterizing additional holograms concurrently can be advantageous for both time-multiplexing and performance. In the future, we intend to conduct further experiments and distribute the allocated point budget across time-multiplexed holograms as well to generate a total number of $n_h \times 3$ holograms simultaneously, with the aim of further enhancing the system's performance. As a direct consequence, the computational cost for the FFT propagation of the WRP would also scale linearly. Nevertheless, we hypothesize that additional avenues for enhancing performance addressing this issue also exist.

Regarding artifacts, the spacing of points has a pronounced effect on the resulting speckle noise, due to the destructive interference between overlapping neighboring disks. Implementing a superior low-discrepancy point distribution could also prove beneficial in this regard. For instance, an effective blue noise method, such as the one proposed by Wolfe *et al.* [WMAMR22], could further augment the quality.

6. Conclusion

We present a system that employs the Point-Based CGH method to generate holograms in real time. Our findings indicate that, despite its suboptimal scaling properties, the Point-Based method can be optimized to meet a vast range of performance requirements, making it a promising candidate for diverse applications, including augmented reality and full-immersion virtual environments. This enables prospective holographic display technologies to capitalize on this method, which is widely regarded as the most flexible among all CGH techniques. This is due to its capacity to facilitate arbitrary dense and accurate sampling of the wave fields in both the spatial and angular domains. Our work is intended to be regarded as an initial step in this direction. In light of our findings, we posit that there is still scope for enhancement, both in terms of quality and performance, such that this method can be a suitable option for future holographic systems.

Acknowledgments

The authors gratefully acknowledge funding by the DFG under Germany's Excellence Strategy within the Cluster of Excellence PhoenixD (EXC 2122, Project ID 390833453), as well as from the DFG Project "Increasing Realism of Omnidirectional Videos in Virtual Reality" (ID 491805996). Open Access funding enabled and organized by Projekt DEAL.

References

[BCKS21] BLINDER D., CHLIPALA M., KOZACKI T., SCHELKENS P.: Photorealistic computer generated holography with global illumination and path tracing. *Opt. Lett. 46*, 9 (May 2021), 2188–2191. doi:10.1364/OL.422159. 2

[BFK22] BINDER N., FRICKE S., KELLER A.: Massively parallel path space filtering. In *Monte Carlo and Quasi-Monte Carlo Methods* (Cham, 2022), Keller A., (Ed.), Springer International Publishing, pp. 149–168. doi:10.1007/978-3-030-98319-2_7. 10

[BL66] BROWN B. R., LOHMANN A. W.: Complex spatial filtering with binary masks. *Applied Optics 6*, 5 (6 1966), 967—-969. doi:10.1364/AO.5.000967. 2

[BO10] BAYRAKTAR M., ÖZCAN M.: Method to calculate the far field of three-dimensional objects for computer-generated holography. *Appl. Opt. 49*, 24 (Aug. 2010), 4647–4654. doi:10.1364/AO.49.004647. 2

[CGP*21] CHOI S., GOPAKUMAR M., PENG Y., KIM J., WETZSTEIN G.: Neural 3D holography: learning accurate wave propagation models for 3D holographic virtual and augmented reality displays. *ACM Trans. Graph. 40*, 6 (Dec. 2021). doi:10.1145/3478513.3480542. 2

[CGP*22] CHOI S., GOPAKUMAR M., PENG Y., KIM J., O'TOOLE M., WETZSTEIN G.: Time-multiplexed neural holography: A flexible framework for holographic near-eye displays with fast heavily-quantized spatial light modulators. In *Conf. Comput. Graph. Interact. Tech. (SIGGRAPH)* (New York, NY, USA, 2022), ACM. doi:10.1145/3528233.3530734. 2, 6

[CW09] CHEN R. H.-Y., WILKINSON T. D.: Computer generated hologram from point cloud using graphics processor. *Appl. Opt. 48*, 36 (Dec. 2009), 6841–6850. doi:10.1364/AO.48.006841. 3

[FCC*24] FRICKE S., CASPARY R., CASTILLO S., EISEMANN M., MAGNOR M.: Modern hardware accelerated point based holography. *Opt. Express 32*, 15 (July 2024), 26994–27009. doi:10.1364/OE.523829. 2, 3, 4, 5, 8, 9, 10

[Gab48] GABOR D.: A new microscopic principle. *Nature 161* (May 1948), 777–778. doi:10.1038/161777a0. 2

[GLC*24] GOPAKUMAR M., LEE G.-Y., CHOI S., CHAO B., PENG Y., KIM J., WETZSTEIN G.: Full-colour 3D holographic augmented-reality displays with metasurface waveguides. *Nature* (2024), 791–797. doi:10.1038/s41586-024-07386-0. 2

[HKH*16] HONG J., KIM Y., HONG S., SHIN C., KANG H.: Gaze contingent hologram synthesis for holographic head-mounted display. In *Practical Holography XXX: Materials and Applications* (2016), Bjelkhagen H. I., Jr. V. M. B., (Eds.), vol. 9771, International Society for Optics and Photonics, SPIE, p. 97710K. doi:10.1117/12.2214274. 2

[HS78] HSUEH C. K., SAWCHUK A. A.: Computer-generated double-phase holograms. *Appl. Opt. 17*, 24 (Dec 1978), 3874–3883. URL: https://opg.optica.org/ao/abstract.cfm?URI=ao-17-24-3874, doi:10.1364/AO.17.003874. 6

[KGC*22] KIM J., GOPAKUMAR M., CHOI S., PENG Y., LOPES W., WETZSTEIN G.: Holographic glasses for virtual reality. In *ACM SIGGRAPH* (New York, NY, USA, 2022), SIGGRAPH '22, ACM. doi:10.1145/3528233.3530739. 1

[KNC*24] KIM D., NAM S.-W., CHOI S., SEO J.-M., WETZSTEIN G., JEONG Y.: Holographic parallax improves 3D perceptual realism. *ACM Trans. Graph. 43*, 4 (July 2024). doi:10.1145/3658168. 1, 3, 4

[KYY08] KANG H., YAMAGUCHI T., YOSHIKAWA H.: Accurate phase-added stereogram to improve the coherent stereogram. *Appl. Opt. 47*, 19 (2008), D44–D54. doi:10.1364/AO.47.000D44. 3

[Lat19] LATYCHEVSKAIA T.: Iterative phase retrieval for digital holography: tutorial. *J. Opt. Soc. Am. A 36*, 12 (Dec. 2019), D31–D40. doi:10.1364/JOSAA.36.000D31. 2

[LKL*22] LEE B., KIM D., LEE S., CHEN C., LEE B.: High-contrast, speckle-free, true 3d holography via binary cgh optimization. *Scientific Reports 12*, 2811 (2022). doi:10.1038/s41598-022-06405-2. 2

[Luc93] LUCENTE M. E.: Interactive computation of holograms using a look-up table. *Journal of Electronic Imaging 2*, 1 (1993), 28 – 34. doi:10.1117/12.133376. 3

[LWHC23] LIU K., WU J., HE Z., CAO L.: 4K-DMDNet: Diffraction model-driven network for 4k computer-generated holography. *Opto-Electronic Advances 6*, 5 (2023), 220135–1. doi:10.29026/oea.2023.220135. 2

[MBS21] MAGALLÓN J., BLESA A., SERÓN F.: SLM simulation and MonteCarlo path tracing for computer-generated holograms. *SN Computer Science 2*, 3 (2021), 1–16. doi:10.1007/s42979-021-00632-6. 2

[MG68] MCCRICKERD J. T., GEORGE N.: Holographic stereogram from sequential component photographs. *Applied Physics Letters 12*, 1 (01 1968), 10–12. doi:10.1063/1.1651831. 3

[PCPW20] PENG Y., CHOI S., PADMANABAN N., WETZSTEIN G.: Neural holography with camera-in-the-loop training. *ACM Trans. Graph. 39*, 6 (Nov. 2020). doi:10.1145/3414685.3417802. 2

[PM03] PETZ C., MAGNOR M.: Fast hologram synthesis for 3D geometry models using graphics hardware. In *Practical Holography XVII and Holographic Materials IX* (2003), Jeong T. H., Stevenson S. H., (Eds.), vol. 5005, International Society for Optics and Photonics, SPIE, pp. 266 – 275. doi:10.1117/12.476879. 2, 3

[PPW19] PADMANABAN N., PENG Y., WETZSTEIN G.: Holographic near-eye displays based on overlap-add stereograms. *ACM Trans. Graph. 38*, 6 (Nov. 2019). doi:10.1145/3355089.3356517. 3

[SBB*22] SHIMOBABA T., BLINDER D., BIRNBAUM T., HOSHI I., SHIOMI H., SCHELKENS P., ITO T.: Deep-learning computational holography: A review. *Frontiers in Photonics 3* (2022), 854391. doi:10.3389/fphot.2022.854391. 2

[SBMS15] SYMEONIDOU A., BLINDER D., MUNTEANU A., SCHELKENS P.: Computer-generated holograms by multiple wavefront recording plane method with occlusion culling. *Opt. Express 23*, 17 (Aug. 2015), 22149–22161. doi:10.1364/OE.23.022149. 3

[SLK*21] SHI L., LI B., KIM C., KELLNHOFER P., MATUSIK W.: Towards real-time photorealistic 3d holography with deep neural networks. *Nature 591* (2021), 234 – 239. doi:10.1038/s41586-020-03152-0. 2

[SMI09] SHIMOBABA T., MASUDA N., ITO T.: Simple and fast calculation algorithm for computer-generated hologram with wavefront recording plane. *Opt. Lett. 34*, 20 (Oct. 2009), 3133–3135. doi:10.1364/OL.34.003133. 3, 6

[SMT*18] SHIMOBABA T., MATSUSHIMA K., TAKAHASHI T., NAGAHAMA Y., HASEGAWA S., SANO M., HIRAYAMA R., KAKUE T., ITO T.: Fast, large-scale hologram calculation in wavelet domain. *Optics Communications 412* (2018), 80–84. doi:10.1016/j.optcom.2017.11.066. 3

[SSMG20] SAHIN E., STOYKOVA E., MÄKINEN J., GOTCHEV A.: Computer-generated holograms for 3D imaging: A survey. *ACM Comput. Surv. 53*, 2 (Mar. 2020). doi:10.1145/3378444. 2

[Wat66] WATERS J. P.: HOLOGRAPHIC IMAGE SYNTHESIS UTILIZING THEORETICAL METHODS. *Applied Physics Letters 9*, 11 (12 1966), 405–407. doi:10.1063/1.1754630. 3

[WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing 13*, 4 (2004), 600–612. doi:10.1109/TIP.2003.819861. 7

[WMAMR22] WOLFE A., MORRICAL N., AKENINE-MÖLLER T., RAMAMOORTHI R.: Spatiotemporal Blue Noise Masks. In *Eurographics Symposium on Rendering* (2022), Ghosh A., Wei L.-Y., (Eds.), The Eurographics Association. doi:10.2312/sr.20221161. 10

[YHHO93] YAMAGUCHI M., HOSHINO H., HONDA T., OHYAMA N.: Phase-added stereogram: calculation of hologram using computer graphics technique. In *Practical Holography VII: Imaging and Materials* (1993), Benton S. A., (Ed.), vol. 1914, International Society for Optics and Photonics, SPIE, pp. 25 – 31. doi:10.1117/12.155027. 3

[ZBA*07] ZIEGLER R., BUCHELI S., AHRENBERG L., MAGNOR M., GROSS M.: A bidirectional light field - hologram transform. *Computer Graphics Forum 26*, 3 (2007), 435–446. doi:10.1111/j.1467-8659.2007.01066.x. 8

[ZCC*11] ZHANG H., COLLINGS N., CHEN J., CROSSLAND B. A., CHU D., XIE J.: Full parallax three-dimensional display with occlusion effect using computer generated hologram. *Optical Engineering 50*, 7 (2011), 074003. doi:10.1117/1.3599871. 4

[ZCZ*17] ZHANG Z., CHEN S., ZHENG H., ZENG Z., GAO H., YU Y., ASUNDI A. K.: Full-color holographic 3D display using slice-based fractional Fourier transform combined with free-space fresnel diffraction. *Appl. Opt. 56*, 20 (July 2017), 5668–5675. doi:10.1364/AO.56.005668. 10

[ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 586–595. doi:10.1109/CVPR.2018.00068. 7