Holoimages

Xianfeng Gu[*]
Computer Science
Stony Brook University

Song Zhang[†]
Mathematics Department
Harvard University

Liangjun Zhang[‡]
Computer Science
UNC-Chapel Hill

Ralph Martin[§]
School of Computer Science
Cardiff University

Peisen Huang[¶]
Mechanical Engineering
Stony Brook University

Shing-Tung Yau[‖]
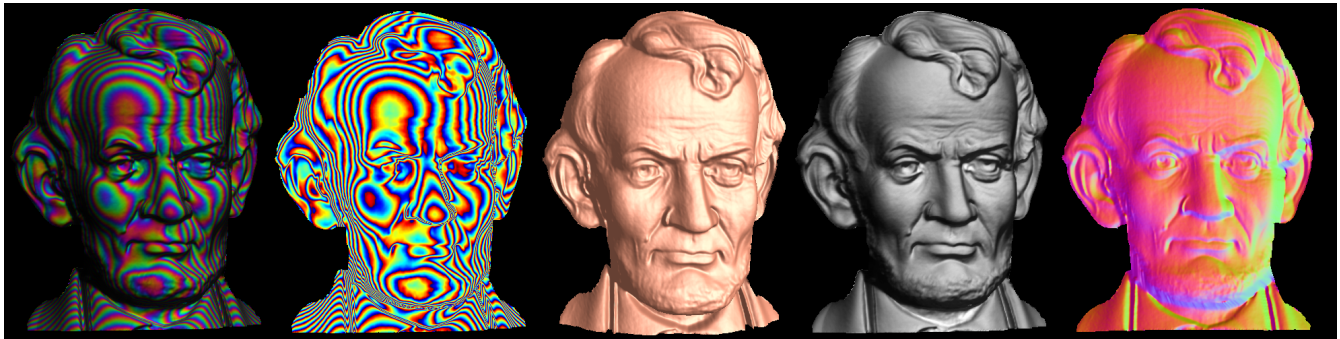Mathematics Department
Harvard University

Figure 1: **Holoimage representing both geometry and shading.** The first image is a 24-bit holoimage with 512×512 resolution, with spatial frequency 64Hz, and $80°$ projection angle. All the other geometry and images are deduced from it, and are, in order, the phase map, the geometric surface, the shaded image, and the normal map.

Abstract

We introduce a novel geometric representation called the *holoimage*, which encodes both shading and geometry information within the same image, based on the principles of wave optics. 'Image' referes to the representation and records the amplitude of the lighting; 'holo' means that it encodes phase, and hence, three-dimensional information. Compared to conventional geometry images or depth images, the holoimage has much higher geometric accuracy. Thus, 3D information can readily be stored and transmitted using the common 24-bit image format.

Holoimages can be efficiently rendered by modern graphics hardware; rendering speed is independent of the geometric complexity and only determined by the image resolution. Rendering holoimages requires no meshes, only textures.

Holoimages allow various geometric processing tasks to be performed simply using straightforward image processing methods, including such tasks such as embossing and engraving, geometric texture extraction, and surface deformation measurement.

Conventional geometric representations, such as meshes, point clouds, implicit surfaces and CSG models, can be easily converted to holoimages using conventional rendering techniques in real time. The opposite process, converting holoimages to geometry in the form of a depth map is accomplished efficiently accomplished by graphics hardware.

Furthermore, holoimages can be easily captured from the real world with a projector and a camera at video frame rate.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling —Curve, surface, solid, and object representations I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms ; I.3.3 [Computer Graphics]: Picture/Image Generation—Digitizing and scanning ;

Keywords: Geometry image, Holoimage, Wave optics, Fringe projection, Phase shifting, Geometric data acquisition.

1 Introduction

Light is an electromagnetic wave with both amplitude and phase. However, human eyes and most cameras can only perceive amplitude, and thus miss the phase information. Traditional computer graphics is based on geometric optics and normally considers amplitude only. The current work differs from traditional image representations, and image formation methods in that it is based on wave optics. It hence emphasizes the phase which conveys geometric information about the scene.

This paper proposes a new geometric representation, the *holoimage*, which is a combination of a traditional image, including shading, texture and silhouettes, with a fringe texture projected by structured light, which encodes phase information. Through the use of a single image, both shading and 3D geometry can be recovered (see Figure 1).

Holoimages have the following advantages, which are valuable for real applications:

- *High Geometric Accuracy.* Compared to conventional geometry images and depth images, holoimages have much higher 3D accuracy. In his work on geometry images [Gu et al. 2002], the first author found it unworkable to use common

[*]e-mail: gu@cs.sunysb.edu

[†]e-mail:szhang@fas.harvard.edu

[‡]e-mail:zlj@cs.unc.edu

[§]e-mail:ralph@cs.cf.ac.uk

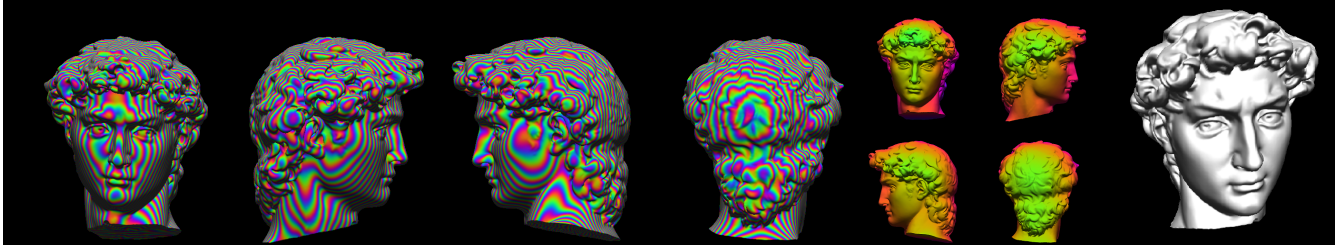[¶]e-mail:peisen.huang@sunysb.edu

[‖]e-mail:yau@math.harvard.edu

Figure 2: **Holoimage-based rendering.** The David head model represented as holoimages viewed from different directions at two wavelengths. The set of holoimages can be efficiently rendered by GPU to cover the whole surface. The last column is the rendered result.

24-bit images to represent a geometry image, as each channel requires at least 12 bits. Holoimages overcome this disadvantage of geometry images. They can be stored and transmitted using 24-bit images. The reason for the higher 3D accuracy is further explained in Section 2.4.

- *Efficient Mesh-Free Rendering.* Holoimages represent both geometry and shading within the same image, allowing them to be rendered efficiently by modern graphics hardware without meshes, using only textures (see Figure 2). This has the potential to simplify the architecture of graphics hardware.

- *High Acquisition Speed.* Holoimages can be captured from the real world with a simple setup involving just a projector and a camera. The reconstruction algorithms are simple enough to be implementable on modern graphics hardware. A sequence of holoimages can be captured at video frame rates for dynamic geometric data acquisition. We have built a system for the capture of dynamic human facial expressions where acquisition and reconstruction are performed in real time at up to 30 frames per second. A description of the system and reconstructed surface data can be found at [Gu et al. 2004].

- *Direct Manipulation.* Holoimages allow image processing techniques to be applied directly to carry out geometric processing, producing such effects as embossing and engraving, geometric texture extraction and deformation measurement. It can be very conveniently converted to and from other geometric representations by hardware automatically (see Figure 10).

Holoimages are based on the principles of wave optics. They inherit some drawbacks from this optical basis:

- *Occlusion.* A holoimage can only represent the part of a surface visible from a single viewpoint. In order to represent the whole surface, several holoimages are required. (Geometry images, however, for example, can represent the whole surface).

- *Special Materials.* Holoimages of dark or glossy materials can not be captured accurately because of their reflectance properties.

The rest of paper is organized as follows: Section 2 introduces the theoretical background, Section 3 reviews related work, Section 4 explains the generation of holoimages in details, Section 5 demonstrates several important real applications, and finally, Section 6 concludes the paper.

2 Background

Light is an electromagnetic wave represented by its electric field. In the special case where a plane wave propagates in the direction
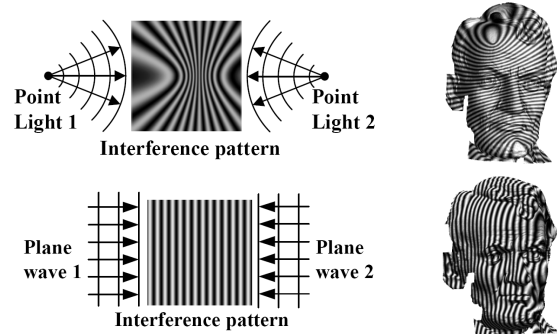

Figure 3: **Principle of holoimages.** Phase information is encoded in the interference fringe pattern. Geometric information is encoded in the phase information.

of a unit vector **n**, the expression describing the field at an arbitrary time t and point with position vector $\mathbf{r} = (x,y,z)$ is given by

$$\psi(\mathbf{r},t) = Ue^{i\phi(\mathbf{r})}e^{-i2\pi vt}, \qquad \phi(\mathbf{r}) = \mathbf{n}\cdot\mathbf{r}/\lambda + \delta,$$

where λ is the wavelength, v the frequency, δ the initial phase, and U the amplitude of the field. The wavefront is parallel to the phase planes satisfying $\mathbf{n}\cdot\mathbf{r} = const$. Since we are interested in the spatial distribution of the field in this research, only the spatial complex amplitude $u = Ue^{i\phi}$ is considered.

There are two general approaches to capturing phase information in image format: interference based methods and fringe projection methods. The latter can be considered to be a special case of the former.

2.1 Interference

Interference occurs when two (or more) waves overlap each other in space. Assume two waves described by $u_1 = U_1e^{i\phi_1}$, $u_2 = U_2e^{i\phi_2}$ overlap. Electromagnetic wave theory indicates that the resulting field is $u = u_1 + u_2$. The intensity is given by

$$I = |u|^2 = |u_1 + u_2|^2 = I_1 + I_2 + 2\sqrt{I_1 I_2}\cos\Delta\phi,$$

where $\Delta\phi = \phi_1 - \phi_2$.

Figure 3 illustrates interference patterns formed by two coherent light sources on a plane, and on a complicated surface. In the first row, the light sources are point lights; in the second row, the light sources are planar lights.

By taking one or more images bearing interference patterns (assuming the surface is not dark or glossy, which would prevent the patterns from being clearly visible) the phase difference $\Delta\phi$ can be estimated with an ambiguity of $2j\pi$ for some integer j. By assuming continuity of the surface, this ambiguity can be eliminated, and the depth map of the objects in the scene can be recovered [Ghiglia
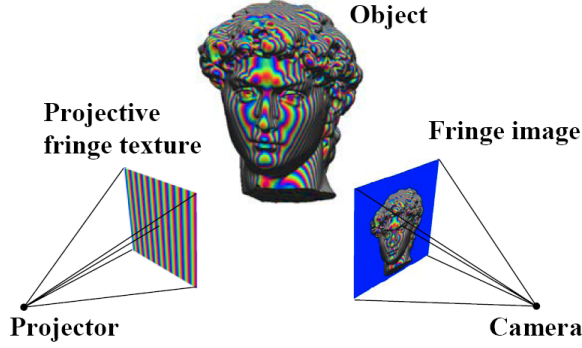
Figure 4: **System setup.**

Figure 5: **Canonical Configuration.**

and Pritt 1998]. The depth map can be converted to geometry once the system is calibrated [Zhang and Huang 2006; **?**]. Interference methods have been widely applied in interferometry and metrology for geometric profile measurement [Gåsvik 2002].

2.2 Fringe Projection

Consider the case shown in the second row of Figure 3. The interference pattern caused by parallel planar light sources can be obtained in an alternative manner using just a single light source of varying intensity, by orthographically projecting a fringe pattern of sinusoidally varying intensity onto the surface in a direction parallel to the wave front as we now show.

If the two planar light sources are located at $x = \pm c$, and that the wave propagation directions are $(\mp 1, 0, 0)$ respectively, then the waves are given by $u_1 = U \exp[2\pi i(x-c)/\lambda]$, $u_2 = U \exp[-2\pi i(x+c)/\lambda]$. The interference wave is thus $u = u_1 + u_2 = 2U \cos[2\pi(x-c)/\lambda]$, and has intensity of the interfering light sd

$$I = 2U^2 \left(\cos[4\pi(x-c)/\lambda] + 1 \right).$$

Thus, instead of making coherent lights interfere on the test surface, the interference fringe pattern can be projected directly to the surface along the z-axis using a planar light with intensity given by the above formula. The fringes can then be viewed from a different angle by a camera. This method is called *fringe projection* [Malacara 1992].

Figure 4 illustrates the basic setup. A sinusoidal fringe pattern is projected onto a geometric object by a projector and an image of the surface with fringes distorted by the geometry is captured by a camera.

Projective Texture We now consider the fringe patterns used to generate a holoimage. A standard computer display projection system may be used to project the fringe pattern onto the object being captured. A texture image is input to the projector. To be able to separate and recover the effects of shape, shading, and ambient light in the fringe patterns, we need to make three independent intensity measurements at each position (x, y), as explained shortly in Section 2.3. We use the red, green and blue channels to store three separate textures to be projected simultaneously; these are 3 sinusoidal fringe patterns with phase differences horizontally in space of $-\frac{2\pi}{3}, 0, \frac{2\pi}{3}$. Each of these ideal sinusoidal fringe textures, having a given spacing, can be described as

$$T_k(u,v) = \frac{1}{2} \left(\cos \left(2\pi u/\lambda + \frac{2(k-1)\pi}{3} \right) + 1 \right), \quad k = 0, 1, 2, \quad (1)$$

where (u, v) are the texture coordinates, λ is the fringe period or inter-fringe distance, k is the channel number. The resulting three-
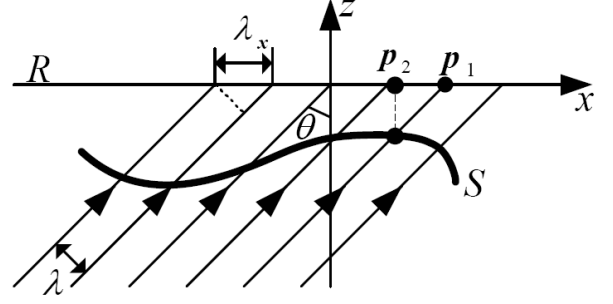
channel holoimage is now captured using a camera, as described next.

Depth recovery from phase We now consider recovery of depth information from the captured fringe patterns, in the form of a depth map $z(x, y)$. We assume a particular geometric setup as follows: the image plane is the xy-plane, a reference plane is at $z = 0$, the optical axis of the camera is the z-axis; the optical axis of the projector is on the xz-plane at an angle θ to the z-axis. The u axis of the projective texture is in the xz-plane; the v axis is parallel to the y axis, so the projected fringes are parallel to the y-axis.

The fringe period in the reference plane along x-axis is thus given by $\lambda_x = \lambda / \cos \theta$. The surface being imaged can be represented as a depth map $z(x, y)$. The intensity at each pixel (x, y) on the fringe image is given by

$$I(x, y) = a(x, y) + r(x, y) \cos(\psi(x, y)). \quad (2)$$

Here, $a(x, y)$ is the *ambient light* intensity, and $r(x, y)$ the *reflectivity* of the surface, which is very similar to the bidirectional reflection distribution function (BRDF) and depends on the direction of lighting, the normal to the surface, and its color, material, and texture. ψ is the *phase* and proportional to the depth $z(x, y)$. A fringe originally positioned at p_1 on a reference plane in front of the object will be displaced to a position p_2 due to the difference in depth between the reference plane and the actual surface (illustrated in Figure 5). The relative depth can recovered using the relationship of

$$z(x, y) \approx \frac{(\psi_2(x, y) - \psi_1(x, y))\lambda}{2\pi \sin \theta}. \quad (3)$$

2.3 Phase Shifting

We measure intensity, and wish to use Equation 2 to recover three unknowns, $a(x, y)$, $r(x, y)$ and $\psi(x, y)$ (and hence depth). Therefore, we need three independent intensity measurements at a given point to do so. We project three separate fringe textures, $k = 0, 1, 2$, with the same wavelength with phase shifts, $-\frac{2\pi}{3}, 0, \frac{2\pi}{3}$, leading to corresponding intensities of

$$I_k(x, y) = a(x, y) + \frac{1}{2} r(x, y)(1 + \cos(\psi(x, y) + \frac{2(k-1)\pi}{3})),$$

The phase ψ, reflectively r, and the ambient light a can, in principle, be computed from these intensities using

$$\psi = \tan^{-1}(\sqrt{3} \frac{I_0 - I_2}{2I_1 - I_0 - I_2}), \quad (4)$$

$$r = 2\sqrt{3(I_0 - I_2)^2 + (2I_1 - I_0 - I_2)^2}, \quad (5)$$

$$a = (I_0 + I_1 + I_2)/3 - r/2, \quad (6)$$
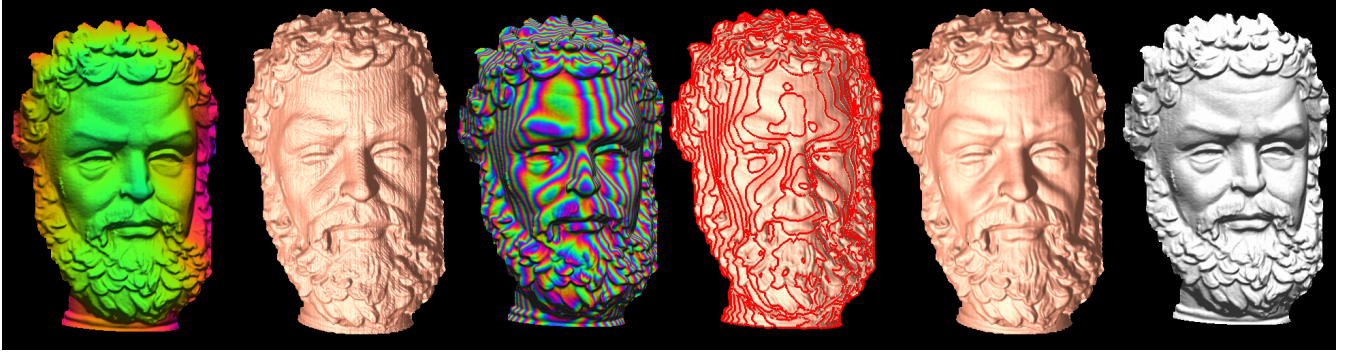
and $a + \frac{r}{2}$ is the reconstructed shading.

Figure 6: **Two-wavelength phase unwrapping.** The first image is a holoimage with spatial frequency 1Hz; the coarse reconstructed geometry is shown in the second image does not need phase unwrapping. This is used as a reference to phase unwrap holoimages with denser fringes. The 3rd image is a holoimage with spatial frequency 64Hz. The 4th image is the reconstructed geometry without phase unwrapping; the red contours show phase jumps. The 5th image is the reconstructed geometric surface after phase unwrapping to depth consistency with the 2nd image. The last image is a shaded image.

Phase Ambiguity   The above process is called *phase wrapping*. The recovered phase using Equation 4 has a 2π phase ambiguity.This phase ambiguity introduces depth jumps on the reconstructed geometry as shown in the 4th image of Figure 6. In order to reconstruct the correct smooth geometry, a *phase-unwrapping* algorithm has be to used. Phase unwrapping intends to remove the artifacts caused by2π discontinuities by adding or subtracting appropriate multiples of 2π.

If the surface is smooth, then the places where phase jump discontinuities occur (the red contours in the 4th image in Figure 6) can easily be located and removed by comparing the phase values with those at neighboring pixels. Although this process is simple, it is unsuited to GPU architecture. Therefore, in our approach, we usually use a two-wavelength method [Creath 1987]. We select 2 spatial periods λ_1, λ_2 for projected textures, such that λ_1 is big enough to cover the whole range of the scene in one period. In this case, there is no phase ambiguity, but the reconstructed geometric accuracy is low. λ_2 is much smaller, and there is phase ambiguity, but the geometric accuracy is much higher. By requiring depth consistency between the two holoimages, the phase of the second one can be *unwrapped*.

Figure 6 illustrates the computation process for the Zeus sculpture with dense geometric features using the two-wavelength method. We also show the Stanford bunny surface with an imposed texture, converted to a holoimage, in Figure 8. Note that both the geometry and the shading with surface texture are reconstructed. Because the original triangulation of the surface is not dense, the flat triangulation structure can clearly be seen on the reconstructed surface. This indicates that the method works robustly for surfaces with complicated textures, and that the geometric resolution of a holoimage can be higher than traditional triangle meshes.

2.4   Error Analysis

The accuracy of the reconstructed geometry is determined by many factors. The direct factors are: the image resolution $m \times m$, the pixel bit-depth n, the fringe spatial frequency λ, and the projection angle θ. Assume there is a one-bit error at a specific pixel in the holoimage. The reconstructed geometric error is

$$\delta z \propto \frac{1}{\lambda 2^n m \sin \theta}. \qquad (7)$$

This formula makes it evident that accuracy can be increased by increasing the projection fringe spatial frequency. In reality, the spatial frequency has an upper limit determined by the resolution of the projector and the resolution of the camera. Similarly, increasing the
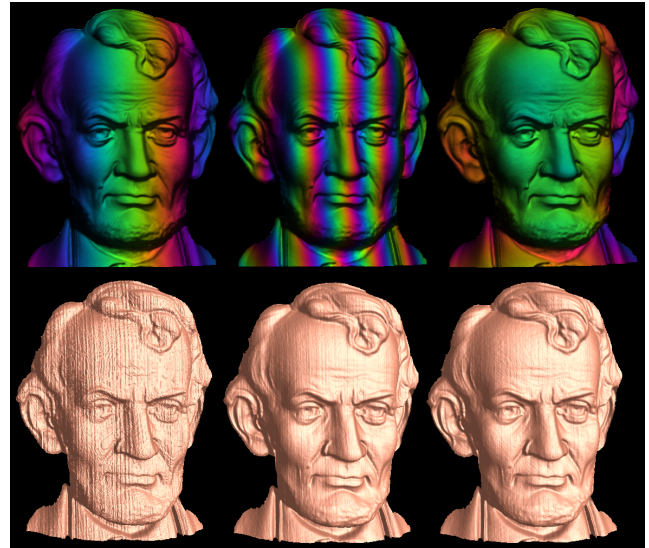

Figure 7: **Holoimage quality is proportional to projection angle and fringe spatial frequency.** The first two holoimages have the same projection angle, 15°, but different spatial frequencies, 2Hz and 8Hz, respectively. The first and the third column holoimages have the same spatial frequency, 2Hz, but different projection angles, 15° and 80°, respectively.

projection angle θ improves the accuracy of the reconstructed geometry but increases the likelihood of occluded regions. The effects of different parameters are illustrated in Figure 7.

Comparison with Geometry Images   Conventional geometry images [Gu et al. 2002] and depth images [Shade et al. 1998] encode the depth information using color information. The geometric error can be formulated as $\delta z \propto 1/m2^n$. Compared with Formula 7, it is obvious that if $\lambda \sin \theta \gg 1$, the geometric accuracy of holoimages is much higher. In practice, we choose λ to be 128, and θ is around 30 degrees. Therefore, holoimage has much higher geometric accuracy than geometry images.

We did some experiments to compare geometry images and holoimages. The Lincoln statute model in figure 1 are converted to both an geometry image and a holoimage. The holoimage is represented as a 24 bits image, and is enough to reconstruct high quality
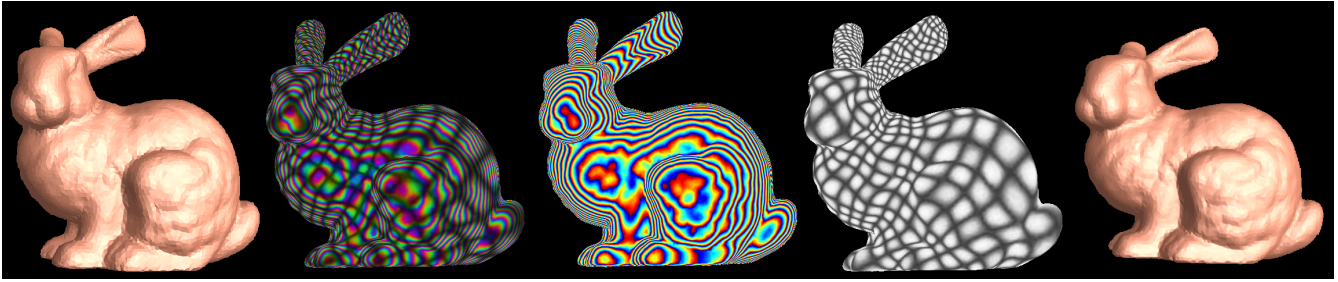
Figure 8: **Holoimage of texture mapped surfaces.** The original geometric surface is flat shaded in the first image, the flat triangles are visible. The second image is the 24-bit holoimage for the textured bunny surface. The third image shows the phase map. The fourth image shows the texture reconstructed from the holoimage. The reconstructed geometry is shown in the last image, the flat triangles are recovered.

surface. But for the geometry image, it requires 36 bits image.

For the stanford bunny surface shown in figure 8,one holoimage can only represent one side of the surface, but a single geometry image can represent the whole surface. Therefore, we use two holoimages to cover the whole surface, the size is comparable to a single geometry image.

3 Related Work

A holoimage is a representation of both geometry and shading in a single image format, which is closely related to the idea of a geometry image. Holoimages can be used to acquire geometric data using simple setups. Thus we should consider the geometric data acquisition literature, and related phase-based methods which have been applied broadly in metrology and interferometry [Gåsvik 2002; Malacara 1992].

3.1 Geometric Representation

Layered depth images are introduced in [Shade et al. 1998], which associate each pixel with a depth value and can be used for efficient rendering.

Geometry images were first introduced in the seminal work of Gu et al. [Gu et al. 2002], which aims to represent geometry using regularly sampled grids. Such grids allow efficient traversal, random access, convolution, composition, down-sampling, compression, and synthesis. Praun and Hoppe generalized geometry images using spherical parameterization [Praun and Hoppe 2003], and applied them to shape compression [Hoppe and Praun 2005]. Smooth geometry images were explored by Losasso et al. in [Losasso et al. 2003], which models general genus zero surfaces by a single spline surface patch. Multi-chart geometry images were introduced by Sander et al. in [Sander et al. 2003] to reduce parametric distortion and improve geometric fidelity. Geometry clipmaps have been utilized for terrain rendering by Losasso and Hoppe in [Losasso and Hoppe 2004].

Geometry images suffer from various disadvantages compared to holoimages. In order to convert conventional irregular meshes to geometry images, sophisticated topological operations must be used, and time-consuming surface parameterization is required. Shading information cannot be encoded in geometry images directly. Furthermore, it is not possible to directly capture geometry images from real objects.

Holoimages are an extension of the idea behind geometry images: representing geometry in an image format. Holoimages can very easily be synthesized in real time using current graphics rendering techniques. The parameter domain is just the image plane, so it is intuitive and thus straightforward to use for editing operations. Shading and texture information can be simultaneously encoded in a holoimage as well as geometry. Furthermore, holoimages can be

directly acquired from real objects using a digital camera and a projector.

3.2 Geometric Data Acquisition

Traditional geometric data acquisition methods are mainly based on geometric optics, where depth information is usually recovered by intersecting two rays. A number of methods have been proposed including stereo, laser stripe scanning, and time or color-coded structured light [Salvi et al. 2004]. Among all existing ranging techniques, stereovision is probably the most studied method. However, the shortcoming of a stereo-based method is that the matching of stereo images is usually time-consuming. It is therefore difficult to realize real-time 3D shape reconstruction from stereo images.

There are essentially two approaches toward real-time 3D shape measurement. One approach is to use a single pattern, typically a color pattern [Harding 1991; Huang et al. 1999]. Because they use color to code the patterns, the shape acquisition result is affected, to varying degrees, by the variations of the object surface color. Rusinkiewicz et al. developed a real-time 3D model acquisition system that utilizes four patterns coded with stripe boundary codes [Rusinkiewicz et al. 2002]. The data acquisition speed achieved was 15 fps. However, like any binary coding method, the spatial resolution is relatively low in comparison with phase-shifting based methods. Huang et al. proposed a high-speed 3D shape measurement based on a rapid phase-shifting technique [Huang et al. 2003]. They use three phase-shifted, sinusoidal grayscale fringe patterns to achieve pixel-level resolution.

Zhang and Huang developed a 3D shape acquisition system with a speed of up to 40 fps [Zhang and Huang 2004]. The geometric shape acquisition, reconstruction, and display was simultaneously realized at 40 fps by using a fast three step phase-shifting algorithm [Huang and Zhang 2006; Zhang 2005]. Moreover, if the system is calibrated by a novel structured light system calibration method proposed by [Zhang and Huang 2006], the 3D geometry measurement error is RMS 0.10-0.22 mm over a volume of of $342(H) \times 376(V) \times 700(D)$ mm. [Zhang and Huang 2004] research works on 3D shape measurement laid the foundations for quickly capturing holoimages from real objects, since recovering the geometry is simple enough to be accomplished in hardware.

3.3 Phase-Dependent Method

Phase-based methods have been broadly applied in engineering for purposes such as optical testing, real time wavefront sensing for active optics, distance measuring using interferometry, surface contouring, and microscopy [Gåsvik 2002; Malacara 1992]. The underlying techniques are based on wave optics. Moiré fringes may be used for the analysis of deformations of materials [Post et al. 1994] as well as for the acquisition of 3D object shapes [Takasaki

1970]. Holoimages can also be generated using traditional Moiré-based methods [Malacara 1992].

4 Generating Holoimages

Holoimages can be easily captured from real objects using structured light, but they can also readily be generated by straightforward rendering methods.

4.1 Generated Holoimages

It is very easy to synthesize a holoimage using a modern graphics pipeline. As exemplified by Figure 4, three sinusoidal fringe patterns can be precomputed and stored as a 3-channel 24-bit color texture image. In order to simplify the analysis of the holoimage, a canonical configuration is preferred, where both the projective texture and the camera use orthogonal projection, and the geometric object is normalized to be inside a unit cube.

In the computation, if we only care about geometric information and do not wish to represent any shading or texture for the surface, we can set the OpenGL texture environment mode to *replace*. If we want to encode both geometry and shading information, we should set the texture environment mode to *modulate*. If a texture is also to be rendered on the surface, we need to use a multitexturing technique to generate the holoimage. Our current methods can only encode monochromatic shading information into a holoimage. If colour shading information is required, we can keep a separately rendered image without any fringe texture for use as a texture image for the reconstructed geometry.

Figure 8 illustrates a holoimage which represents geometry shaded with a complicated texture. The holoimage was synthesized using the following procedure: first, we parameterized the Stanford bunny surface using conformal parameterizations (any parameterization method could have been used), then we texture mapped an arbitrary monochromatic bitmap onto it, and finally we shaded color sinusoidal fringes on the bunny using projective texture mapping. These two textures were combined using multitexturing. Specifically, we used the *ARB_multitexture* extension of OpenGL on a Nvidia GeForce 6800 graphics card, setting the texture environment mode to modulate, to blend the surface texture color and the fringe pattern.

4.2 Captured Holoimages

To capture holoimages from real objects, we use a Digital-Light-Processing (DLP) projector (Plus U2-1200) to generate the fringe patterns, and a high-speed digital CCD camera (Dalsa A-D6-0512) synchronized with the projector so that we can capture each projected color channel separately. A DLP projector is a digital projector that has a unique projection mechanism make it brighter than a LCD projector. Since its digital nature, it makes the contrast of the projected image larger than a LCD projector. Moreover, a single chip DLP projector projects three color channels sequentially, we take advantages of this projection mechanism and realized real-time 3-D measurement. By sequentially capturing RGB channels of the projected image, we realize the functionality of a color fringe pattern while eliminating color-matching problems.

Data can be acquired at a speed limited by those of the projector (120Hz) and camera used; in practice we can achieve 30Hz. This is adequate to capture general moving surface deformations, such as human expressions. Geometric 3D data illustrated in Figures 1, 6, 14, 15, and 16 were captured from real objects using our system. Photographs of our system, further captured holoimages, and reconstructed surfaces are available at [Gu et al. 2004].
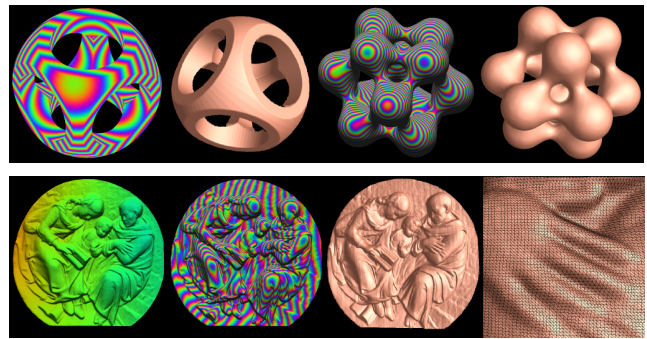

Figure 10: **Conversion.** Other geometric representations can be easily converted to holoimages using projective texture. The top row shows conversion from a CSG model and an implicit surface. The bottom row shows conversion from a point cloud.

4.3 Differences

There are fundamental differences between a synthesized holoimage and one captured from real life: the projective texture mapping of a synthetic holoimage does not include shadows or self-occlusion. Therefore, the projection angle can be as large as a right angle to improve the accuracy of the reconstructed geometry. For captured holoimages, in order to avoid shadows caused by projector, the projection angle must usually be quite small.

Another difference is related to color. Three monochromatic fringe projective textures can be combined into one color projective texture and a color 24-bit holoimage can be synthesized. However, using color holoimages for geometry capture is undesirable since the measurement accuracy is affected by the color of the object. Therefore, three monochromatic fringe images are usually needed to reconstruct the geometry.

5 Applications

Holoimages can be easily produced from other geometric representations as shown in Figure 10. Holoimages are valuable for many important applications.

Most surface models in this paper are captured by our holoimage system, such as 1,6,7,15,16,14 and the second row of 10.

5.1 Mesh-free Rendering

A holoimage represents both shading and geometry within the same image. Recent development of graphics hardware makes it feasible to efficiently render holoimages directly using its programmable pipeline. For rendering purposes, we believe holoimages have the potential to replace traditional meshes, and therefore, simplify the architecture of GPU.

5.1.1 Surface Reconstruction using Holoimages

The architecture of our surface reconstruction algorithm can be readily mapped on to graphics processors, promising a significant acceleration over CPU based methods. The surface reconstruction algorithm requires computationally intensive and accurate numerical computations, on a complete grid pattern, and fully satisfies the requirements for an SIMD computation [Bolz et al. 2003; Purcell et al. 2002]. Therefore, our reconstruction algorithm can be efficiently accelerated on high performance graphics processors (GPUs), such as nVIDIA GeForce 6800 and ATI Radeon 9800, which expose a flexible SIMD programming interface with powerful floating computational capacity.
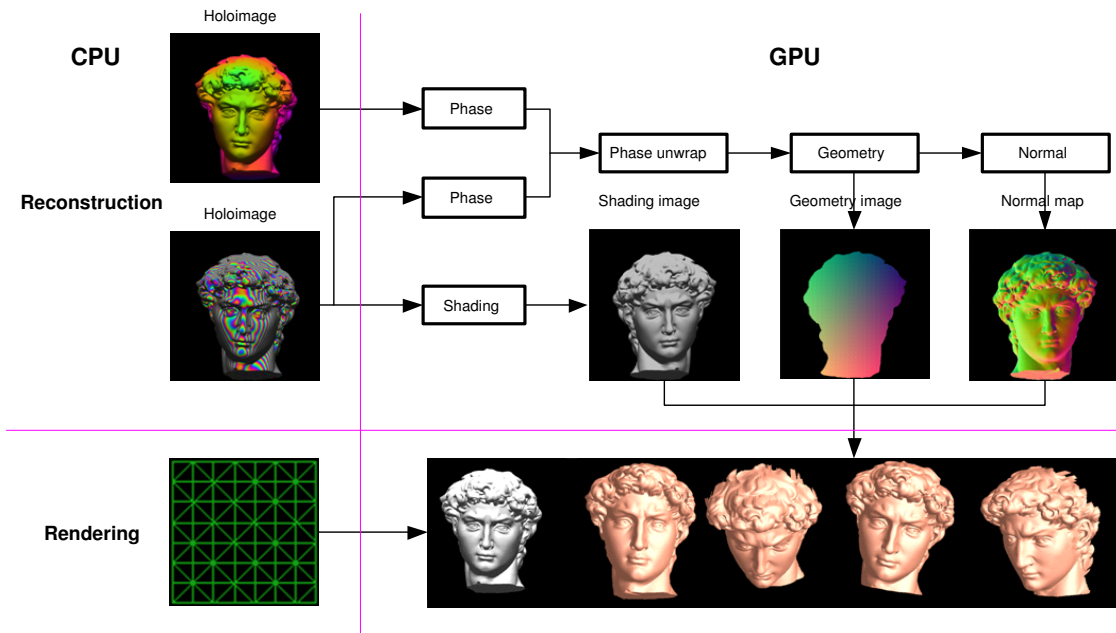
Figure 9: **Pipeline of surface reconstruction and rendering for holoimages on GPU.**
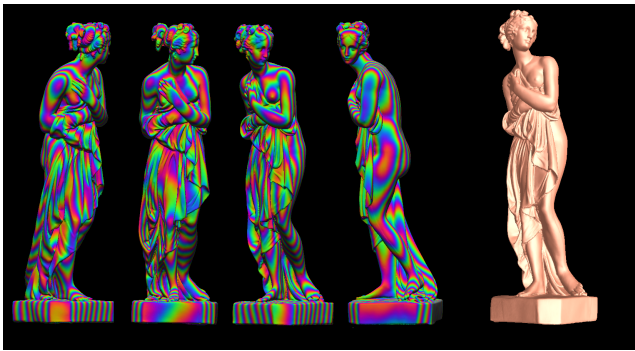

Figure 11: **Holoimage-based Rendering.**

We map our surface reconstruction to graphics hardware as in Figure 9. The original holoimages are first uploaded to the GPU as textures. Then, a series of GPU computation-intensive kernels are used, based on *pixel shaders*, including a *Phase Kernel* which computes Equation 4, a *Shading Kernel* for Equations 6 and 5, a *Phase Unwrap Kernel*, a *Geometry Kernel* for Equation 3, and a *Normal Kernel*, are sequentially applied to the original and intermediate generated textures. Each computational kernel processes every texel in parallel and produces exactly one element of the intermediate texture for the next processing stage. The results of our reconstruction algorithms, including the geometry image, the shaded image, and the normal map, remain in the GPU memory for further rendering if desired.

Besides performing computation on a regular grid, the algorithm also has the attractive property of local memory access. None of the computational kernels here need to access any adjacent texels, apart from the normal kernel, which only needs to access the 8-connected neighbors. Because streaming hardware can effectively hide the latency of memory access, our GPU-based algorithm shows distinct advantages over the CPU-based version [Purcell et al. 2002].

5.1.2 Holoimages Rendering

Holoimages which have just been captured can be rendered efficiently because they do not need to transferred from GPU to CPU and back for rendering, thus overcoming a common performance bottleneck within GPU accelerated systems [Carr and Hart 2004]. To render the recovered geometry, the system constructs a dummy grid mesh in the GPU, with the same connectivity as the object surface but fake geometric and shading information. The texture coordinates assigned to each input vertex encode its grid indices. The *vertex shader*, which on current GPUs can directly access the texture memory, then fetches the true geometry position, shading color, and normal vector for each vertex using the pre-assigned texture coordinates, and computes the transformed position and lighting color. In practice, the rendering can be further optimized by drawing indexed primitives, which can reduce the number of executions of vertex shaders.

Adaptive real-time rendering can easily be performed using holoimages, allowing a smooth transition between texture and geometry. When the rendered object is far away from the synthetic camera, the system can simply map the shading texture to a rectangle. When the object gets closer, the system adds the normal maps to the textured rectangle. Finally, once the object is close enough, the geometry is rendered using the texture and normal maps.

5.1.3 Performance

We have implemented our GPU-based reconstruction and rendering algorithms; Figure 11 shows a further example of rendering, while Figure 12 shows times taken by various steps of our algorithm for CPU and GPU based implementations, at different resolutions. The overall time includes the time required for each kernel and system overheads, which include the time for texture loading and system setup. The GPU-accelerated system was implemented using OpenGL and the Cg Toolkit to do the high-level GPU programming. The computation kernel for reconstruction was implemented as a pixel shader. To fully utilize the channels for each texel, the phase kernel and shading kernel were merged, and only one intermediate texture was generated. All tests were performed on a 3.00GHz Intel Processor, with an nVIDIA GeForce 6800 graphics card.
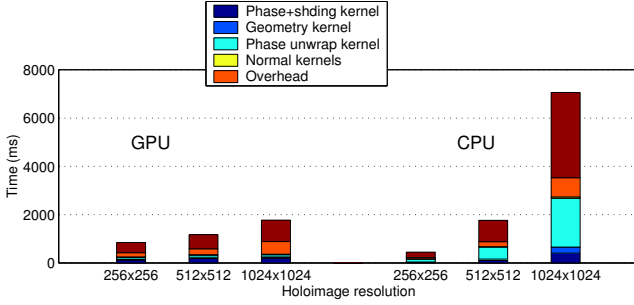
Figure 12: **Performance comparison for GPU-based and CPU-based surface reconstruction.**

For input holoimages with 256×256 resolution, the CPU-based version is faster. However, for larger holoimages, the GPU-based implementation significantly outperforms the CPU-based method. Typically, to reconstruct a surface (200K faces) from high-resolution (1024×1024) holoimages, the GPU-based system is much more efficient than the CPU implementation. Figure 12 indicates that the greatest savings are made when using graphics hardware during normal kernel compuations, due to the GPU-based normal kernel using built-in hardware vector operations, such as cross product and normalization.

Figures 2 and 6, showing David and Zeus, demonstrate that our reconstruction algorithms are insensitive to complexity of the input geometry.

Moreover, our experiments showed that within each kernel, only a small percentage of time is occupied by arithmetic instructions, while most of the time is used for texture lookups. This suggests our GPU-based reconstruction algorithm may be further accelerated.

Our efficient GPU-based methods for surface reconstruction and rendering based on holoimages leads us to believe that by representing geometry, shading and normal information of surfaces with images, holoimages may be used to replace triangle meshes for rendering purposes. Also, holoimages have the potential to simplify the architecture of modern graphics hardware.

5.2 Geometric Processing

By using holoimages to represent geometry on a regular grid, many geometric processing tasks can be accomplished by image processing techniques, which can also be accomplished efficiently by graphics hardware.

Since geometry is encoded in the phase information, geometric processing can be performed using phase information directly. Suppose the phase of a holoimage is $\psi(x,y)$. A *phase map* is a map from the image plane to the unit circle, $\sigma : R^2 \to S^1, \sigma(x,y) = e^{i\psi(x,y)}$. Two phase maps can be composed together by pixel-wise multiplication or division. Then the set of phase maps forms a group

$$\Sigma = \{e^{i\psi(x,y)}\}, \sigma_1 \circ \sigma_2(x,y) = \sigma_1(x,y)\sigma_2(x,y).$$

By composing the phase maps of two surfaces, it is very easy to generate embossing and engraving effects using GPU processing of holoimages.

In our implementation, we find it is sufficient to use only 16 bits per image to represent a phase map. The red and blue channels are used to encode the real and the imaginary parts of the phase map,

$$r(x,y) = 128\cos\psi(x,y) + 128, \quad b(x,y) = 128\sin\psi(x,y) + 128$$

Then the wrapped phase $\psi(x,y)$ can be simply computed as

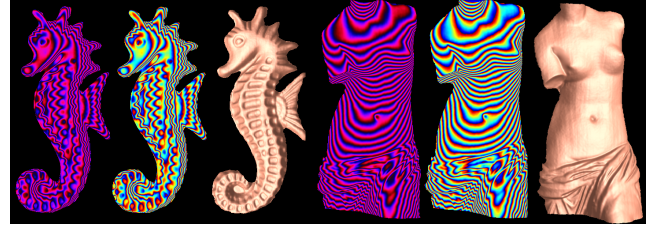$$\psi(x,y) = \tan^{-1}\frac{b(x,y) - 128}{r(x,y) - 128}.$$
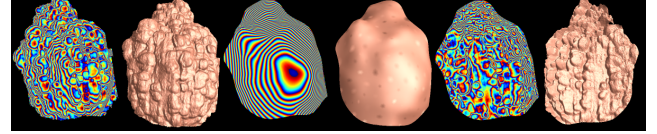

Figure 14: **16-bit phase maps.**

Figure 15: **Geometric texture extraction.** The images from left to right are an original holoimage and associated geometry, holoimage and smoothed geometry, extracted texture holoimage and geometry.

Figure 14 illustrates 16-bit holoimages and the corresponding geometries for the Seahorse and Venus surfaces.

Embossing and Engraving Embossing is implemented by multiplying two phase maps. The following algorithm uses the following steps for the embossing process:

1. Choose the camera position and zoom factor for the two surfaces.

2. Render the holoimages of the two surfaces with two wavelengths.

3. Compute the wrapped phase of each holoimage and convert them to phase maps.

4. Multiply the corresponding phase maps of different surfaces.

5. Compute the new unwrapped phase using the two-wavelength algorithm.

6. Convert the phase function to a geometric surface.

All the operations in the algorithm can be performed pixel-wise without requiring information from neighboring pixels, and can thus be readily implemented on a GPU. Figures 13 illustrates examples of our embossing and engraving algorithm.

Geometric Texture Extraction Textures are important for geometric modeling. Although it is easy to capture image textures, it is generally difficult to capture geometric textures. It is desirable to extract geometric texture from real surfaces with complicated profiles and transfer the texture to other geometric models.

By using holoimages, we can extract geometric textures from real objects easily. First a complex surface is scanned and the geometry reconstructed. By using conventional geometric smoothing techniques, we remove the geometric texture from the surface. Then, by subtracting the smooth surface from the original surface, the geometric texture can be extracted. The process is illustrated in Figure 15.

Deformation Measurement It is useful to detect and measure deformation of surfaces for many applications, for example to model human facial expressions and muscle movements. Figure 16 shows an interesting application of holoimages, 'a smile without a face'. We captured a human face with different expressions. Two
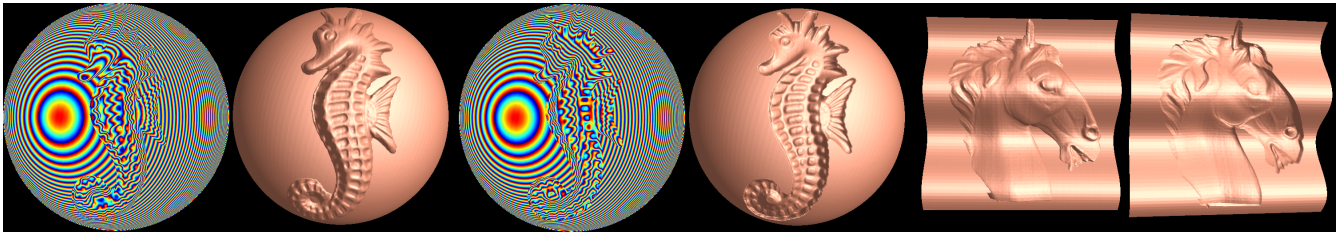
Figure 13: **Geometry editing using holoimage.** The seahorse surface is used to make an embossed or engraved relief on a sphere, showing phase maps and geometric results. Further images show embossing and engraving of a horse's head on a wave surface.
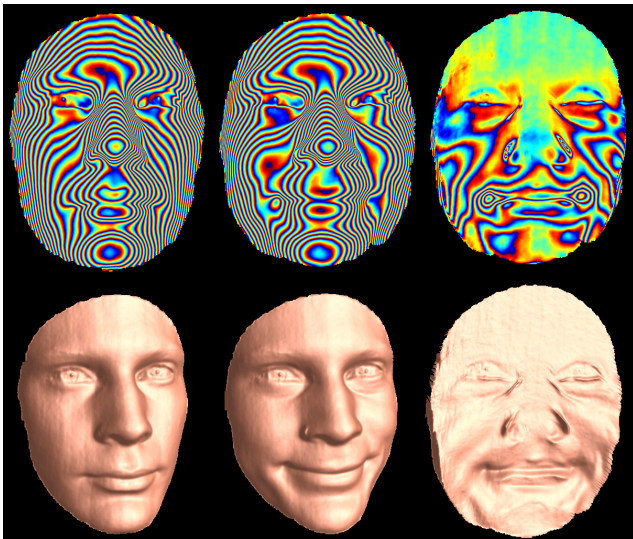

Figure 16: **Distorsion measurement by holoimage.** The phase maps and the geometric surfaces of the calm face, the smiling face and the smile itself are shown. On the smile phase map, the distortion is proportional to the density of the fringes. All the subtleties of the smile are accurately measured.

frames were selected, one face being neutral, the other with a smile. By converting both of them to phase maps, computing the difference phase map, and reconstructing the geometry, 'a smile without a face' can be obtained. From the deformation phase map, it is clear that the lips of the mouth have changed the most, the muscles on the cheeks have changed moderately, and the forehead and the nose bridge have remained unchanged.

6 Conclusions

This paper has introduced a novel geometric representation called the *holoimage*, which encodes both amplitude and phase information in a single image. Compared to conventional geometry images and depth images, holoimages have much higher geometric accuracy, and can be stored and transmitted using common image formats.

Holoimages can be synthesized using a conventional graphics pipeline or captured from real life with a simple hardware setup, in both cases using fast algorithms. Furthermore, the algorithms to reconstruct a depth map from a holoimage are simple and can be accomplished by graphics hardware.

Holoimages can be efficiently rendered using GPU hardware. Holoimage based rendering is mesh-free and has the potential to lead to simpler GPU architecture.

Holoimages can be applied to a range of geometric process-

ing tasks using image processing methods, again utilizing graphics hardware. These include embossing, engraving, geometric texture extraction and distortion measurement.

In the future, extending holoimage based methods to better handle occlusions will be explored. Acquiring holoimages from glossy or dark surfaces will also be investigated.

Acknowledgement

References

BOLZ, J., FARMER, I., GRINSPUN, E., AND SCHRÖDER, P. 2003. Sparse matrix solvers on the gpu: conjugate gradients and multigrid. In *Proceedings of SIGGRAPH 2003*, ACM Press / ACM SIGGRAPH, 917–924.

CARR, N. A., AND HART, J. C. 2004. Painting detail. In *Proceedings of SIGGRAPH 2004*, ACM Press / ACM SIGGRAPH, 845–852.

CREATH, K. 1987. Step height measurement using two-wavelength phase-shifting interferometry. *Appl. Opt. 26*, 2810–2816.

GÅSVIK, K. J. 2002. *Optical Metrology*, 3rd ed. John Wiley and Sons, Inc.

GHIGLIA, D. C., AND PRITT, M. D. 1998. *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software*. John Wiley and Sons, Inc.

GU, X., GORTLER, S., AND HOPPE, H. 2002. Geometry images. In *Proceedings of SIGGRAPH 2002*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 355–361.

GU, X., ZHANG, S., AND HUANG, P. 2004. University of new york at stony brook geometric surface archive. http://metrology.eng.sunysb.edu/holoimage/index.htm.

HARDING, K. G. 1991. Phase grating use for slop discrimination in moiré contouring. In *Proc. SPIE*, vol. 1614 of *Optics, illumination, and image sensing for machine vision VI*, 265–270.

HOPPE, H., AND PRAUN, E. 2005. Shape compression using spherical geometry images. In *Advances in Multiresolution for Geometric Modelling*, N. Dodgson, M. Floater, and M. Sabin, Eds., 27–46.

HUANG, P. S., AND ZHANG, S. 2006. A fast three-step phase shifting algorithm. *Appl. Opt. (under press)*.

HUANG, P. S., HU, Q., JIN, F., AND CHIANG, F. P. 1999. Color-encoded digital fringe projection technique for high-speed three-dimensional surface contouring. *Opt. Eng. 38*, 1065–1071.

HUANG, P. S., ZHANG, C., AND CHIANG, F. P. 2003. High-speed 3-d shape measurement based on digital fringe projection. *Opt. Eng. 42*, 1, 163–168.

LOSASSO, F., AND HOPPE, H. 2004. Geometry clipmaps: terrain rendering using nested regular grids. *ACM Trans. Graph. 23*, 3, 769–776.

LOSASSO, F., HOPPE, H., SCHAEFER, S., AND WARREN, J. 2003. Smooth geometry images. In *Eurographics Symposium on Geometry Processing*, 138–145.

MALACARA, D., Ed. 1992. *Optical Shop Testing*. John Wiley and Songs, NY.

POST, D., HAN, B., AND IFJU, P. 1994. *High-Sensitivity Moiré: Experimental Analysis for Mechanics and Materials*. Springer-Verlag, Berlin.

PRAUN, E., AND HOPPE, H. 2003. Spherical parametrization and remeshing. *ACM Trans. Graph. 22*, 3, 340–349.

PURCELL, T. J., BUCK, I., MARK, W. R., AND HANRAHAN, P. 2002. Ray tracing on programmable graphics hardware. In *Proceedings of SIGGRAPH 2002*, ACM Press / ACM SIGGRAPH, 703–712.

RUSINKIEWICZ, S., HALL-HOLT, O., AND LEVOY, M. 2002. Real-time 3d model acquisition. *ACM Trans. Graph. 21*, 3, 438–446.

SALVI, J., PAGES, J., AND BATLLE, J. 2004. Pattern codification strategies in structured light systems. *Pattern Recogn. 37*, 4, 827–849.

SANDER, P. V., WOOD, Z. J., GORTLER, S. J., SNYDER, J., AND HOPPE, H. 2003. Multi-chart geometry images. In *Eurographics Symposium on Geometry Processing*, 146–155.

SHADE, J. W., GORTLER, S. J., WEI HE, L., AND SZELISKI, R. 1998. Layered depth images. In *Proceedings of SIGGRAPH 98*, 231–242.

TAKASAKI, H. 1970. Moiré topography. *Appl. Opt. 9*, 6, 1467–1472.

ZHANG, S., AND HUANG, P. 2004. High-resolution, real-time 3-d shape acquisition. In *IEEE Computer Vision and Pattern Recognition Workshop (CVPRW'04)*, vol. 3, 28–37.

ZHANG, S., AND HUANG, P. S. 2006. A novel structured light system calibration. *Opt. Eng. (under press)*.

ZHANG, S. 2005. *High-Resolution, Real-Time 3-D Shape Measurement*. PhD thesis, Stony Brook University, State University of New York.