A hardware architecture for rapid generation of electro-holographic fringe patterns

John A. Watlington, Mark Lucente, Carlton J. Sparrell, V. Michael Bove, Jr.

{wad, lucente, carltonj, vmb}@media.mit.edu

MIT Media Laboratory

Ichiro Tamitani

tamitani@dsp.cl.nec.co.jp

Information Technology Research Labs, NEC Corporation

ABSTRACT

This report describes the hardware architecture and software implementation of a hologram computing system developed at the MIT Media Laboratory. The hologram computing employs specialized stream-processing hardware embedded in the Cheops Image Processing system – a compact, block data-flow parallel processor. A superposition stream processor performs weighted summations of arbitrary one-dimensional basis functions. A two-step holographic computation method – called Hogel-Vector encoding – utilizes the stream processor's computational power. An array of encoded hogel vectors, generated from a three-dimensional scene description, is rapidly decoded using the processor. The resulting 36-megabyte holographic pattern is transferred to framebuffers and then fed to a real-time electro-holographic display, producing three-dimensional holographic images. System performance is sufficient to generate an image volume approximately 100 mm per side in 3 seconds. The architecture is scalable over a limited range in both display size and computational power. The limitations on system scalability will be identified and solutions proposed.

Keywords: computer-generated holography, electro-holography, holovideo, real-time holographic displays, parallel processing, dataflow processing, intelligent framebuffers

1 INTRODUCTION

Optical holography[1] uses the physical phenomena of interference and diffraction to record and reconstruct a three-dimensional (3-D) image. A holographic fringe pattern ("fringe") diffracts light because its feature size is generally on the order of the wavelength of visible light (about 0.5 micrometer). A computer-generated hologram (CGH)[2] represents a fringe pattern as an array of discrete samples. The sampling rate (or pitch) must be high enough to accurately represent the holographic information – typically about 2 micrometer^{-1}. Because of this microscopic resolution, a numerically computed fringe pattern contains an enormous number of samples – over

20,000 samples/cm on a single line of the hologram ("hololine").

A real-time electro-holographic ("holovideo") display[3] requires the computation of fringe patterns containing millions of samples. The size and complexity of a holographic fringe pattern often precludes computation at interactive rates. Even with the power currently available in scientific workstations, researchers in the field of computational holography commonly report computation times of minutes or hours.

One way to increase the speed of fringe computation is to reduce the sample count required in a holographic fringe. The elimination of vertical parallax typically reduces sample count by a factor of about one hundred, and allows for each hololine to be computed and displayed independently. The 2-D holographic pattern representing an HPO 3-D image can be treated as a vertical array of 1-D holograms or hololines.[4]

A second way of increasing the speed is by reducing the complexity of the operation performed. This report presents a hardware architecture for implementing Hogel-Vector decoding, a novel algorithm based on the recently reported Diffraction-Specific fringe computation.[5] The use of Hogel-Vector encoding reduces computational complexity. The holographic display system, which has been described previously in the literature,[3] is briefly introduced in Section 2. The theoretical underpinnings of the computational algorithm used to generate the fringe patterns for the display are presented in Section 3. The electronic system driving the display is presented in Section 4, and the special purpose processor designed to implement the computation algorithm in Section 4.2. Experimental results are then presented in Section 5, along with future directions of research.

2 HOLOGRAPHIC DISPLAY

A real-time holographic display uses computed fringes to modulate a beam of light and produce an image. The heart of a holographic display is the spatial light modulator (SLM) used to modulate light with a computed fringe pattern. An ideal holographic SLM does not yet exist, but time-multiplexing of a very fast SLM provides a suitable substitute. To display images using the holographic fringes computed on Cheops, we used a display system that combines an 18-channel acousto-optic modulator (AOM) and a series of lenses and scanning mirrors to assemble a real 3-D holographic image at video frame rates.[6] A general description follows, and a detailed description can be found in the reference by St. Hilaire.[6] We remark that the holographic computation method described in this paper is not specific to one display. By incorporating the proper physical parameters and sample size, a hologram generated using this method can be viewed using other holographic displays.

In our display, (see Figure 1) 18 parallel signals are supplied from a memory buffer storing the fringe pattern to a pair of cross-fired AOMs. The 144 holoines contain a total of 37,748,736 samples (36 MB). As each hololine of the fringe pattern is read out of the memory, it is converted to an analog signal and passes through a radio frequency signal processing circuit into one channel of the AOM. Each of the 18 parallel channels has a data rate of 110 MSamples/s, providing each a signal bandwidth of 55 MHz, and an aggregate display bandwidth of 990 MHz. Each channel is represented in memory by an 8-bit sample, giving an output SNR of 59 dB. At any instant, as 18 lines of the holographic pattern traverse the aperture of the AOM in the form of acoustic waves, a portion equal to roughly 1,000 samples (in each channel) modulates the phase of the wavefront of laser light that passes through each channel of the AOM. Two lenses image the diffracted light at a plane in front of the viewer. By reflecting the light off of a synchronized horizontally scanning mirror system, the apparent motion of the holographic pattern is canceled. The scanning mirror system also angularly multiplexes the image of the acoustic wave. It extends the apparent width of the imaged holographic pattern to 262,142 samples, each sample representing a physical spacing of 0.58 micrometer. Currently, the display limits the usable fringe spectrum to a range from 0 to 0.4 cycles/sample (0 to 690 cycles/mm).

The display produces a real 3-D image located just in front of the output lens of the system. The image occupies a volume that is 135 mm wide, 85 mm high and 150 mm deep. The width of the viewing zone –

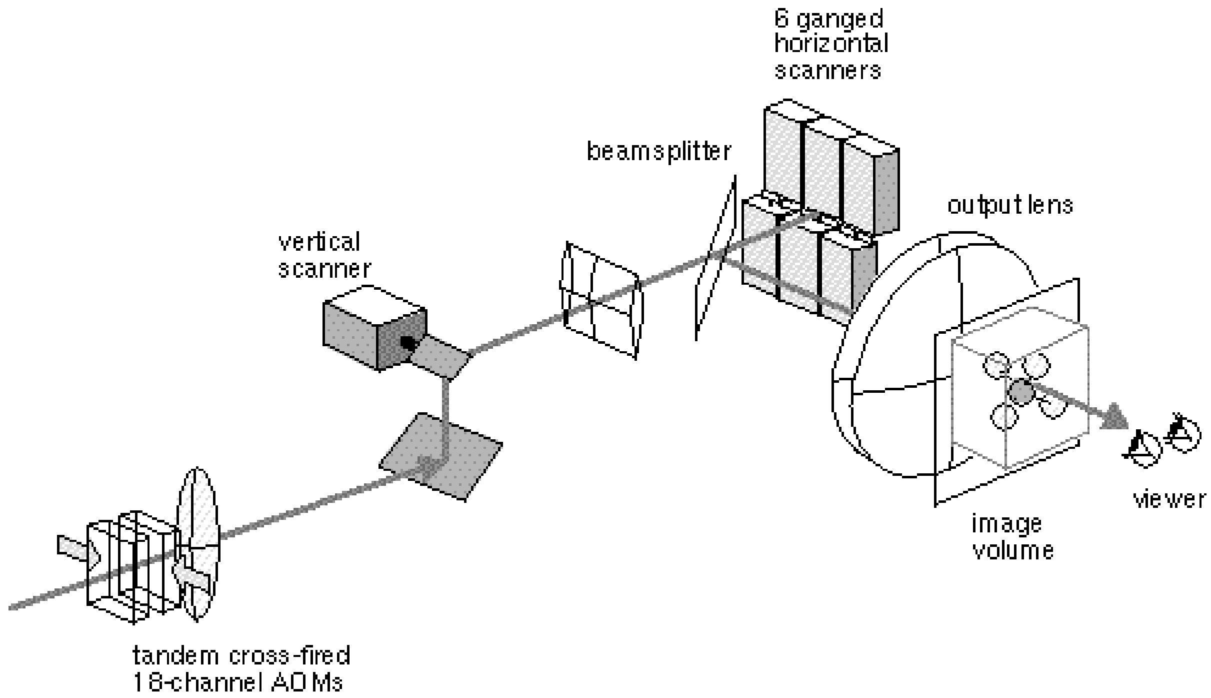

Figure 1: 36-MB Holovideo Display: Partial Schematic of the holovideo display used in this research. The horizontal scanning mirror system angularly multiplexes the image of the modulated light in a back-and-forth motion. A vertical scanning mirror vertically positions each set of 18 hololines. Electronic control circuits synchronize the scanners with the incoming holographic signal.

i.e., the range of eye locations from which the viewer can see the image – is 30 degrees horizontal. This is an HPO display. Because the holographic image possesses no vertical parallax, there is no need for diffraction in the vertical dimension. The vertical resolution of 1.7 lines/mm is equivalent to that of a 19" diagonal NTSC television receiver. Each hololine diffracts light to a single horizontal plane to form image elements describing a horizontal slice of the image. Therefore, one hololine needs to contain only the contributions from elements that lie on a single horizontal slice of the object.

3 FRINGE COMPUTATION TECHNIQUE

Computational holography[2] generally begins with a 3-D numerical description of the object or scene to be reproduced. Traditionally, computational holography was slow due to two fundamental properties of fringe patterns: (1) the enormous number of samples required to represent microscopic features, and (2) the computational complexity associated with the physical simulation of light propagation and interference. Interactive computation has been achieved[7] using optimized algorithms implemented on a 16,384 processor Connection Machine 2 massively parallel supercomputer (CM-2). In recent work,[5] Lucente used a new approach called Diffraction-Specific fringe computation to calculate holographic fringes at nearly interactive rates using simple computational hardware. The next subsection briefly describes Diffraction-Specific computation and its corollary Hogel-Vector holographic bandwidth compression algorithm, and is followed by a detailed description of the algorithm that

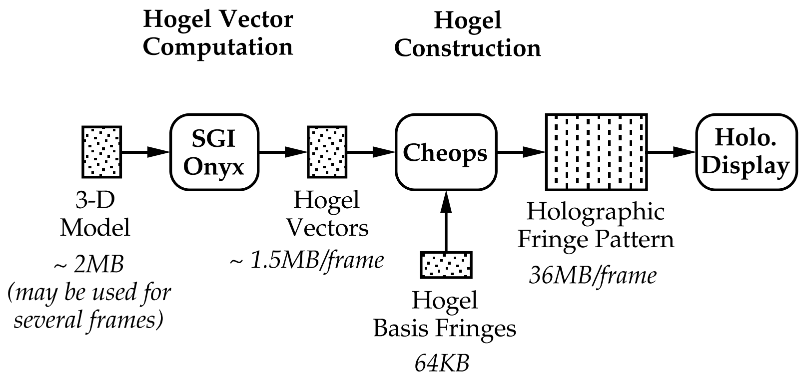

Figure 2: Overview of hogel-vector generation and decoding

was implemented on the Cheops system.

3.1 Diffraction-Specific fringe computation

Diffraction-Specific fringe computation[5] is based on the discretization of space and spatial frequency in the fringe pattern. Diffraction-Specific computation was invented to satisfy two primary goals: (1) to produce fringes with a reduced amount of computation and (2) to enable holographic representations which reduce the bandwidth required to transmit holographic images. In Diffraction-Specific computation, the fringe is treated as an array of regularly spaced holographic elements called "hogels." Each hogel – typically 0.5 mm in width – has a homogeneous spectrum. Each hogel's spectrum is discretized into N evenly spaced regions. For a given hogel, the array of coefficients that describe the energy in the discrete spectral regions is called a "hogel vector."

Diffraction-Specific computation encompasses two types of holographic bandwidth compression techniques. This paper focuses on Hogel-Vector encoding, which is a two-step process, as shown in the Figure 2. First, the array of hogel vectors is computed from the description of the 3-D scene to be imaged using a ray-tracing algorithm akin to standard computer graphics techniques. A hogel vector in a particular location in the hologram plane contains the information necessary to calculate the fringe pattern for that one hogel. The second step is to convert the array of hogel vectors into an array of hogels, *i.e.*, the actual fringes. In this conversion ("decoding") step, each hogel-vector component is used as a weighting factor in the linear summation of a set of precomputed elemental fringes – called "basis fringes." (See Figure 3.) Each basis fringe is specially computed to contain energy in a particular region of the spectrum. Therefore, by weighting each basis fringe with its corresponding hogel-vector component and summing the results, the resulting hogel contains the spectral energy specified by the hogel vector. The construction of a hogel from a hogel-vector is then an inner product between the hogel vector and the array of basis fringes. The specific parameters used in our experiments are discussed in section 5.1.

4 CHEOPS: MODULAR IMAGE PROCESSING SYSTEM

The non-standard display format and the large display bandwidth made difficult the use of commercially available electronic display systems to provide the signals for the holographic display system. Instead, an image processing system, Cheops, developed for research use at the MIT Media Lab, was extended to support the holographic display. The Cheops system was capable of meeting the required display parameters, and also provided a scalable modular platform for experimenting with algorithm acceleration. A small submodule containing a super-

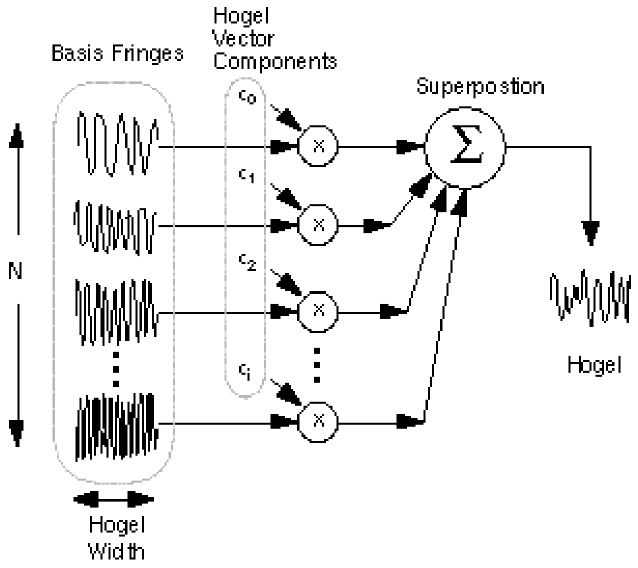

Figure 3: The linear superposition of basis fringes constructs ("decodes") a hogel from a hogel vector.

position processor (the Splotch Engine)[8] was then designed and built to obtain interactive computation speeds in holovideo using the Hogel-Vector method. Only the system features relevant to the holographic configuration will be discussed, as other descriptions of Cheops have been published.[9]

Figure 4 shows a block diagram of a Cheops system configured for driving the holographic display system. Six output modules (O1), each supplying three channels, are used to generate the eighteen signals output to the holographic display. One, or possibly two, processor modules (P2) manipulate and load data into the output modules. An optional memory module (M1) provides a small amount (1 to 4 GBytes) of storage local to the system. These modules are interconnected by two linear buses. One of these buses is capable of sustained high-bandwidth (120 MByte/s) transfer of sample rasters (the Nile Bus) and the second is a 32 MByte/s bus for transfer of smaller units of data and system control (the Global Bus). A second Nile Bus provided by the system is not supported by the output modules, but may be used by the processor and memory modules.

The primary function of the output modules is to decouple the display data rate (30 frames/s) from the computation and data transfer rate (0.1 to 3 frames/s, for the holovideo application). Each module contains three 8-bit channels, each with 2 MBytes of memory, configured to allow simultaneous reading for updating a scanned display and writing over the Nile bus. These are very similar to the output modules used by Cheops systems configured for "normal" high-resolution video. The sole differences are an ability to synchronize their output scanning and sample clocks, and support for horizontal line sizes up to 262,142 samples in length.[10]

4.1 Processor module architecture

The philosophy behind the design of the Cheops processor module is to allow computationally intensive operations to be abstracted out and embodied in specialized hardware provided with a decoupled very high-throughput memory interface, under the control of a general-purpose processor. Despite the large size of the datasets manipulated in image processing, they are usually accessed in a regular fashion. Rather than requiring

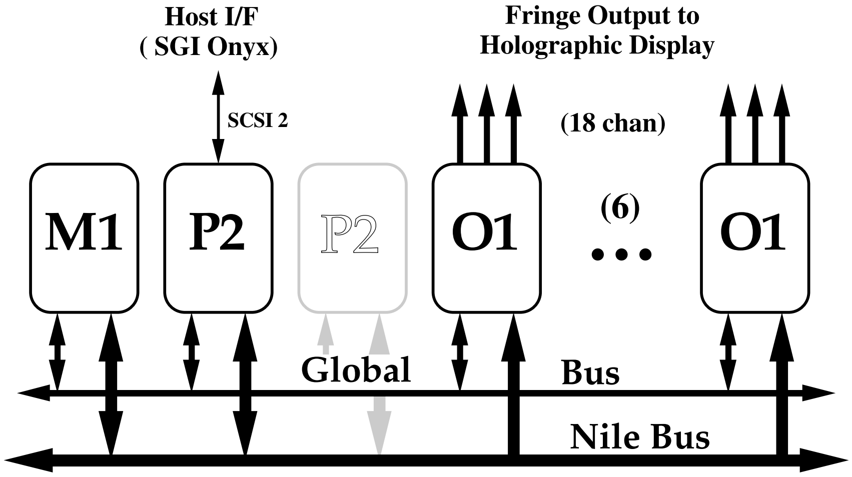

Figure 4: Cheops system overview

that each specialized processor provide a large amount of local storage, or providing it with a high-speed random access path to the main processor memory, the processors operate upon one or more high-speed streams of data.

The processor module comprises eight memory units communicating through a full crosspoint switch with up to eight stream processing units. Figure 5 shows a simplified view of the topology of a single "P2" processor module. Each memory unit is made up of 4 MBytes of dual-ported dynamic memory (VRAM) and a two-dimensional memory access controller (a "flood controller") for transferring a stream of data through the crosspoint at up to 32 MSample/s (each sample is 16 bits). The flood controllers are capable of relatively agile handling of one- and two- dimensional (and to a lesser extent, three-dimensional) arrays, but are limited to managing only one stream at a time. Up to a total of four processing pipelines, consisting of stream sources, processors, and destinations may occur simultaneously on a single processor module.

Specialized stream processors designed and built for the Cheops system perform common mathematical tasks such as convolution, correlation, matrix algebra, block transforms, spatial remapping, or nonlinear functions. These processing units are on removable submodules, allowing reconfigurability and easy upgrade. Two stream input and two stream output ports are provided per submodule. If a stream processor (*e.g.*, the Splotch Engine) requires multiple input or output streams, it may utilize the entire submodule and occupy two processor slots.

A general-purpose 32-bit central processing unit – an Intel 80960CF, clocked at 32 MHz – is provided on the processor module for sequencing and controlling the flow of data among the different functional units, implementing the portions of algorithms for which a specialized processor is not available, and performing higher-level tasks such as resource management and user interface. A resource manager[11] is provided both to allow multitasking and to enable the applications programmer to obtain parallelism with minimum effort. A dataflow representation of the algorithm is provided to the resource manager, which is then responsible for executing the algorithm with the maximum parallelism possible, given the current configuration of the system and the other concurrent tasks.

The processor module communicates with a host computer using a Small Computer Systems Interface (SCSI-2) bus. Each processor module appears as a fixed disk device to the host, allowing transfer of application code and data at rates of 1.5 MByte/s. An application which performs hogel-vector decoding typically receives hogel vectors and basis fringes from the host computer over this SCSI-2 interface. An interface between the M1 local storage module and host systems using the High Performance Parallel Interface (HiPPI) bus, which will allow a data transfer rate of 100 MByte/s, is currently being integrated into the system.

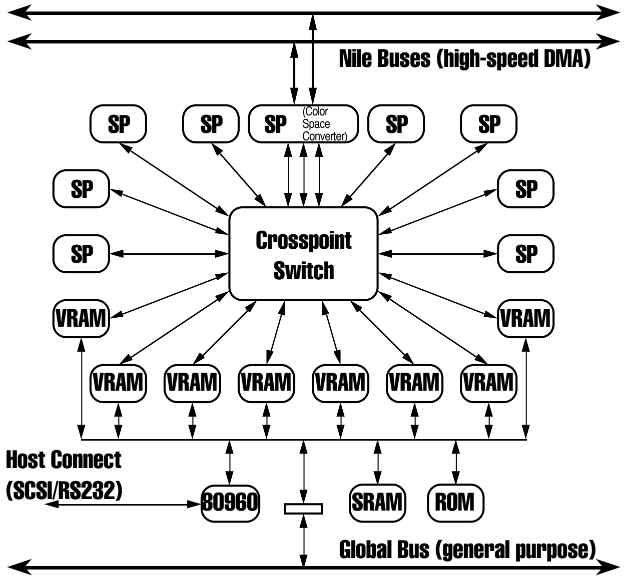

Figure 5: Block diagram of the Cheops P2 processor module. Blocks labeled SP represent specialized stream processors, while those labeled VRAM are banks of dual-ported memory equipped with high-speed DMA controllers.

4.2 The superposition stream processor

The operation required for constructing the hogels from the hogel vectors is a matrix inner product. Given the sometimes sparse nature of the hogel vectors – in some areas of the fringe pattern there are as few as eight nonzero components in the hogel vector – we specialized the operation to be a superposition of select one-dimensional basis functions. Review of the superposition operation shows that two of the data accesses – the read of original data and the writing of the summed data – have a data-independent memory access pattern, and are thus suitable for implementation using streams.

The simplest implementation of a stream superposition processor (a “superposition element”) uses a single multiplier to scale the samples of a basis fringe fetched from a local memory and a single adder to add the scaled basis-fringe samples into a data stream passed through the processor. Both the multiplier and the adder perform one operation per clock cycle, the same rate at which data is transferred by the streams. Multiple passes of the data through such a superposition element, or multiple superposition elements chained together, allow the construction of hogels from hogel-vector arrays of arbitrary size. Multiple superposition elements may be combined in a single stream processor, up to the limiting size of the maximum number of nonzero components in the hogel vectors. Parallelism beyond this is achievable by processing separate hogels simultaneously, at the cost of additional source and destination data streams.

The actual dynamic range required by the holographic fringe patterns is a subject of investigation. Since the Cheops system was originally designed for manipulating video, whose SNR rarely exceeds 59 dB, the Nile Bus and output module data paths are 8 bits/channel. This was adopted as the dynamic range of the fringe patterns output to the holographic display system. Internal to the P2 processor module, signals are represented in memory by signed or unsigned 16-bit words. The operating precision of the Splotch Engine was chosen to be 16-bit unsigned words, although operating modes that specify the basis function and its scaling factor using only 8 bits are provided, allowing for twice as many basis functions,

The limiting factors in the number of superposition elements contained in a single Splotch Engine were the physical form factor of a Cheops stream processor (10cm $\times$ 14cm) and the large package sizes of the integrated

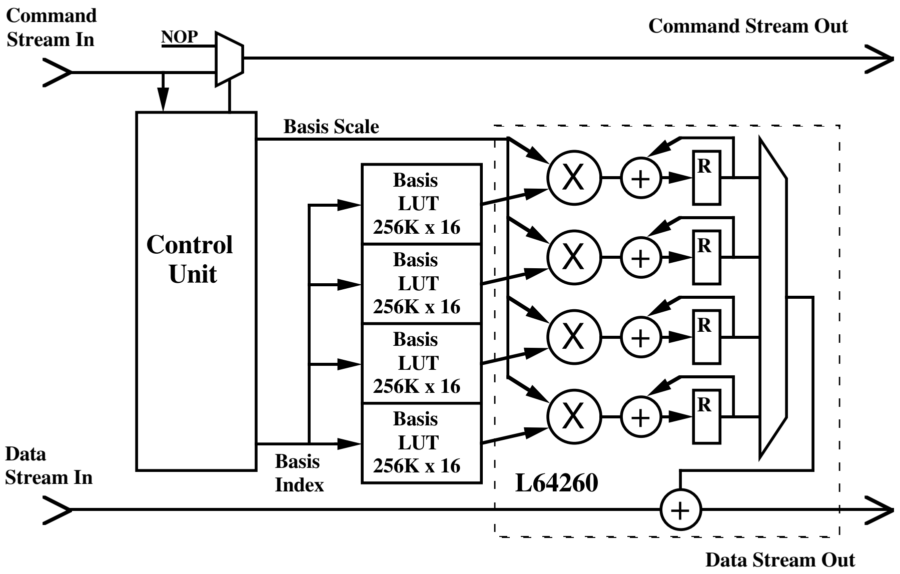

Figure 6: Cheops Splotch Engine block diagram

circuits (IC) available commercially. In 1992 (when design on the Splotch Engine was begun) it was possible to include four superposition elements on a single stream processor submodule, along with common control logic. A block diagram is shown in Figure 6. The basic components of the design are a computational block, the basis function (fringe) sample memory, and a control unit. In order to greatly simplify the control unit, the Splotch Engine only supports 1-D basis functions in hardware.

A control buffer containing "splotch commands" is used to specify the operations to be performed. Null, or nop, commands are used to fill unused locations in the command buffer. A stream generated by reading this buffer is input to the Splotch Engine in parallel with the data being superpositioned. Each command corresponds to a single nonzero component of a hogel vector, *i.e.*, the superposition of a single basis fringe. The control buffer has the same size as the data being superpositioned, and the origin of the basis fringe is determined by the position of the command in the command buffer. Basis functions may be co-sited by storing shifted versions of a basis function in the memory of different superposition elements. Upon encountering a command in the control stream, a Splotch Engine attempts to execute it. If it has a superposition element idle, it accepts the command and replaces it in the control stream with a nop. If it cannot accept the command, it leaves it in the command stream. A hardware flag readable by the resource manager records if any commands could not be executed, allowing a single command buffer to be processed iteratively until completely executed.

Multiple splotch command instruction formats are defined, allowing different ranges of parameters. However, only one command format is supported at any time, selectable by a control register. Two of the command definitions fit in a 16-bit word, allowing 256 basis fringes with an 8-bit scaling value or 512 basis fringes with a 7-bit scaling value. A third command definition allows up to 64K basis fringes to be used with a 16-bit scaling value, but requires that two 16-bit samples be used to represent the command. The number of samples in a basis fringe is also defined by a control register, to be one of { 16, 20, 32, 36, 256, 512, 516, 1024 }.

The basis memory consists of one bank of 256K $\times$ 16 bits of static RAM per superposition element. The basis fringes are constant, so they are downloaded into the superposition processor during system initialization. The size of the basis function memory was determined by the width of the memory datapath (64 bits, providing four 16-bit basis samples) and the size of fast static RAM modules readily available at the time of design. In addition, if the desired operation is a block basis transform (*i.e.*, all basis functions are superpositioned starting at a single

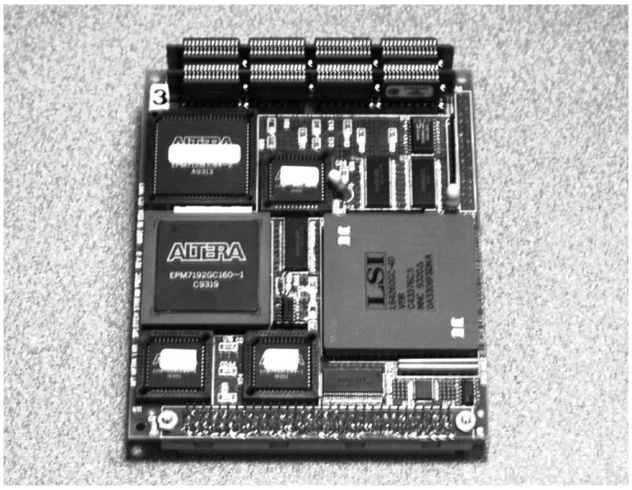

Figure 7: A Splotch Engine stream processor

location) the basis functions may be distributed among the superposition elements in a single Splotch Engine, instead of each superposition element having a complete copy of the basis functions.

The computational block was implemented by an L64260 IC from LSI Logic. It provided four 16-bit multipliers and four 40-bit accumulators capable of running at up to 40 MHz, as well as the input and output datapaths (6 $\times$ 16 bits) required by the design. Unfortunately, it was available only in a large 224-pin grid array package.

Like the individual superposition elements, the Splotch Engines may be chained serially. A single Cheops P2 processor module may be configured with up to three Splotch Engines, allowing chains with four, eight, or twelve superposition elements to be built. The level of parallelism available is smaller than that presented by constructing a typical test image, where the maximum number of nonzero hogel-vector components is typically 32. Multiple passes of the entire hololine of data through the Splotch Engines are therefore required to completely construct the hogels from the hogel vectors.

5 RESULTS

Hogel-Vector decoding has been implemented on different computer systems, providing a performance comparison. The holovideo Cheops system described above was the only one capable of directly driving the holographic display system with the computational results. The parameters of the fringe computation, as described next, were largely determined by the holographic display subsystem.

5.1 Fringe computation parameters

Diffraction-Specific fringe computation began with the selection of hogel width. Based on the relationships described in the reference by Lucente,[5] we chose a hogel width of 1024 samples (a physical width of 0.6 mm) This allowed for reasonably sharp images and a factor of 64 reduction in holographic bandwidth and in calculations per fringe sample. A set of 16 basis fringes, represented as 8-bit integers, were then computed using a method of iterative spatial-spectral constraints followed by a simulated annealing treatment.[5] These 16 basis fringes represented $N = 16$ evenly spaced regions of the spatial frequency spectrum, each centered at $0.4 \times ((n+0.5)/N)$

Machine	Computation Time	Effective MOPS	Relative Speedup
SGI Onyx	150 sec.	4.0	1x
CM-2	10.9 sec.	55	14x
IBM PVS	2.5 sec.	242	60x
Cheops, one proc	6 sec.	100	25x
Cheops, two proc	3 sec.	200	50x

Table 1: Holographic fringe computation results

cycles/sample, where $n = \{0, 1, 2, \ldots, N-1\}$. A fixed basis fringe set was used for constructing all hogels.

The hologram fringe pattern was treated as an array of 256×144 (hor. $\times$ ver.) hogels, for a total of 262142×144 samples. In the first step of Hogel-Vector encoding, the 16-component hogel vector for each hogel was generated by geometric optics calculations. This step of the computation used a "diffraction table"[5] that contained a list of the hogel-vector component contributions corresponding to a specific image element. The diffraction-table was indexed by image-element locations in the 3-D image volume. The array of hogel vectors was tabulated from the list of component contributions for each image element. Each hogel vector in the resulting 256×144 hogel-vector array contained 16 8-bit integer components. This hogel-vector array was processed into a compact format for storage and transmittal to the second "hogel-construction" step, either running on one of the commercial systems tested or the holovideo Cheops system.

5.2 Experimental results

The experimental results comparing hogel-vector decoding times for a 36-MB fringe pattern are shown in Table 1. In this table, MOPS refers to a million 16-bit read/multiply/accumulate operations per second. Each hogel required up to 16 basis-fringe superpositions, for a total of 600 million operations. The architecture presented in this report required six seconds for computation. This time was faster than the time achieved when the same algorithm was implemented on the CM-2 massively parallel supercomputer. When two Splotch Engines were used in tandem, computation time was reduced to only three seconds. This represented a calculation rate of 200 million operations per second. The Cheops system has the definite advantage that it connects to the output cards over the high-speed Nile Bus, requiring only a fraction of a second to transfer the 36-MB fringes. The CM-2 or PVS supercomputers require a time-consuming data-transfer step that adds at least three additional seconds to the total compute time.

Although a computation time of three seconds seems slow for interactivity, these timings were performed to illustrate the worst cases. Image scenes of arbitrary complexity require the same amount of time. However, some time savings can result when simpler images are used. The hogel-vector array representing a typical image scene contains many zero values. Therefore, the number of basis-fringe superpositions is reduced, typically by a factor of three, resulting in a reduction in computation time by a factor of three.

Fringes computed on Cheops were used to generate 3-D images on the holovideo display system. (See Figure 8.) As predicted by the principles of Diffraction-Specific fringe computation,[5] the 0.6-mm hogel-width and the use of 16 basis fringes leads to a noticeable but tolerable blur in the deepest image elements.

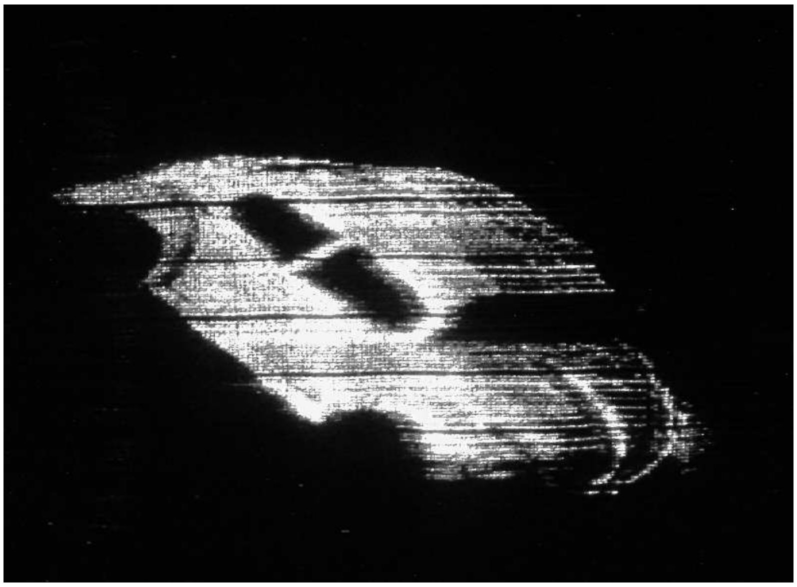

Figure 8: A photograph of a 3-D image (the body of a car) produced on the MIT holovideo display, generated using Diffraction-Specific fringe computation. The 36-MB fringe pattern was computed using the two-step Hogel-Vector encoding technique.

5.3 Future Directions

The ability to transmit and store the large data sets required for computer generated holograms has always presented a significant challenge to developing a reasonable sized holovideo display. Our work with the Cheops based holovideo system has shown that it is possible to transmit encoded fringes and construct a full fringe data set using a pipeline of superposition stream processors. Future work in this area will involve increasing the processing power through parallelism, reducing the data bottleneck, and implementing additional holographic bandwidth compression in the hardware.

The current MIT holovideo system[12] was designed to be scalable. This scalability is based in parallelism in most aspects of the display such as the Frame Buffers, the AOM channels and the scanning mirrors. To provide data rates required for such a scalable system, the superposition computation must also be done in parallel. The current Cheops system will allow increased parallelism through the introduction of a second P2 Processor card. Beyond this, we must look at providing processing that is more integrated in the output phase of the pipeline. To this end, we are planning an intelligent frame buffer, the "Fringebuffer."

As described earlier, a number of superposition stream processors can be chained together to provide suitable processing power without sacrificing throughput or affecting the width of the data path. Increasing the parallelism of the processing in the existing Cheops system would eventually be limited by the bandwidth available between the processor modules and the output modules. To avoid this "data bottleneck", we propose that superposition stream processors be integrated into the output frame buffers, creating an array of intelligent Fringebuffers. This implementation would allow the data stream for each output channel to be constructed as it is being written into the frame buffer memory, utilizing the parallel memory and data paths already in place at that stage. It would also allow progressive hogel construction, where images are rendered using a reduced number of basis fringes when the update rate is critical, yet allow the data in the frame buffers to be processed in additional passes, superpositioning more basis fringes, when the image remains static. This configuration provides a variable real-time trade-off between image quality and the image update rate.

Another plan for holovideo stream processing is the addition of specialized hardware for other holographic encoding methods, such as Fringelet encoding as developed by Lucente.[5] The superposition stream processor can be used to generate a "fringelet" for each Hogel that can be post-processed by a Fringelet decoding stream processor. The lengths of the basis functions used for a fringelet generation is an order of magnitude less than the length of a hogel, reducing the number of multiply-accumulations needed to generate a holographic image.

6 ACKNOWLEDGMENTS

Due to the ongoing nature of both the Cheops and Holovideo project the number of people who have contributed exceeds the limits of the publication guidelines. We do, however, wish to extend our special thanks to the following (in no particular order): Pierre St.-Hilaire, Ravikanth Pappu, Derek Arias, John D. Sutter, Stephen A. Benton, Wendy J. Plesniak, Michael Halle, Shawn Becker, Brett Granger, and numerous undergraduate participants in MIT's Undergraduate Research Opportunities Program.

Components of this research have been sponsored by the Television of Tomorrow research consortium of the Media Laboratory, MIT; Honda Research and Development Co., Ltd.; NEC Corporation; International Business Machines Corp.; and the Advanced Research Projects Agency (ARPA) through the Naval Ordnance Station, Indian Head, Maryland (under contract No. N00174-91-C0117).

7 REFERENCES

[1] P. Hariharan, Optical Holography. Cambridge: Cambridge University Press, 1984.

[2] W. J. Dallas, "Computer Generated Holograms," in The Computer in Optical Research, B. R. Frieden editor, Vol. 41 of Springer Series Topics in Applied Physics. Berlin: Springer Verlag, 1980, pp. 291-366.

[3] P. St.-Hilaire, S. A. Benton, and M. Lucente "Synthetic aperture holography: a novel approach to three dimensional displays," Journal of the Optical Society of America A, vol. 9, no. 11, pp. 1969-1977, Nov. 1992.

[4] D. Leseberg and O. Bryngdahl, "Computer-generated rainbow holograms," Applied Optics, vol. 23, no. 14, pp. 2441-2447, 15 July 1984.

[5] M. Lucente, "Diffraction-Specific Fringe Computation for Electro-Holography," Ph. D. Thesis, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Sept. 1994.

[6] P. St.-Hilaire, "Scalable Optical Architectures for Electronic Holography," Ph. D. Thesis, Program in Media Arts and Sciences, Massachusetts Institute of Technology, Sept. 1994.

[7] M. Lucente, "Interactive computation of holograms using a look-up table," Journal of Electronic Imaging, vol. 2, no. 1, pp. 28-34, Jan 1993.

[8] I. Tamitani, "Splotch Stream Processor", MIT Media Lab internal memo, Aug. 1993.

[9] V. M. Bove, Jr., J. A. Watlington, "Cheops: A Reconfigurable Data-Flow System for Video Processing," to be published in IEEE Trans. on Circuits and Systems for Video Technology, 1995.

[10] D. Arias, "Design of an 18 Channel Frame Buffer for Holographic Video," S.B. Thesis, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Feb. 1992.

[11] I. J. Shen, "Resource Manager for a Video Processing System," SM thesis, MIT, 1992.

[12] Additional information on holovideo research at the MIT Media Laboratory can be found at the following URL address: `http://lucente.www.media.mit.edu/people/lucente/holo/holovideo.html`