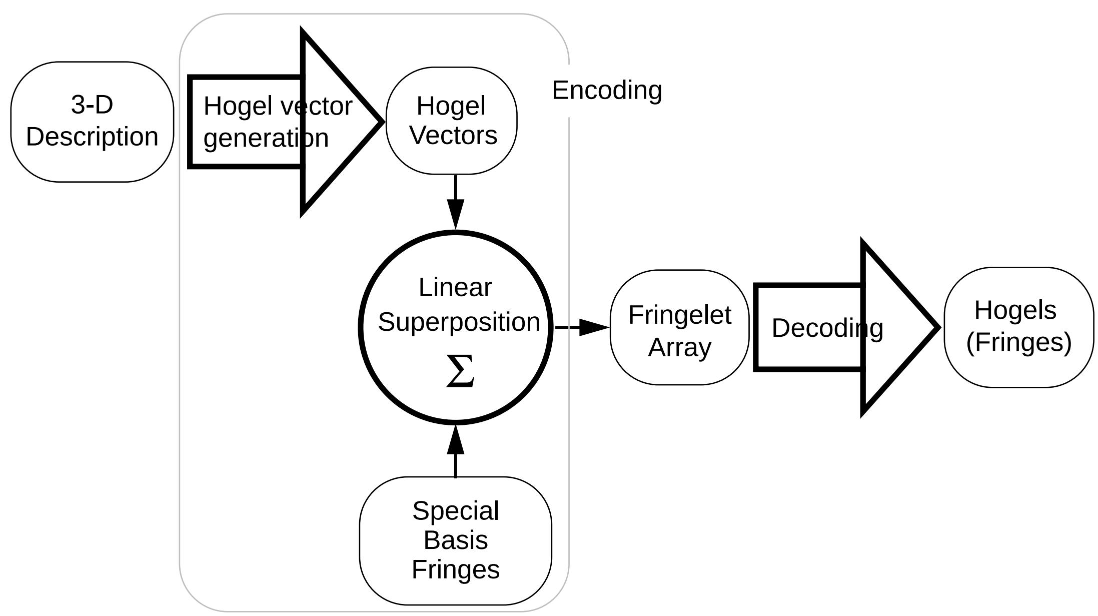
Chapter 6

Fringelet Encoding

The major drawback to using hogel-vector encoding is that the decoding step requires an enormous amount of computing power. A second type of encoding scheme - “fringelet encoding” - is designed to decrease decoding time by using an encoded format that more closely resembles the final fringe pattern. In fringelet encoding, the encoded format for a given hogel is called a *fringelet* because it looks like a small fringe, with a spectrum that is closely related to the desired hogel spectrum. An array of fringelets is computed directly from the set of hogel vectors. The resulting fringelet array is subsequently decoded into usable fringes using an aperiodic replication scheme that is extremely fast. The decoding step is fast because a fringelet contains the sample values that are to appear in a hogel. Decoding involves no arithmetic, only sample replication and reordering.

Overview of Fringelet Encoding and Decoding



Essentially, fringelet encoding and decoding (see overview, above) substitutes for the conversion of hogel vectors to hogels. Each hogel vector is converted into a fringelet - an encoded description of the hogel, but one that occupies only a fraction of the bandwidth (i.e., the number of samples). Fringelets are converted to hogels in the fringelet decoding step. The following sections describe the encoding and decoding steps, including a description of the special spectral qualities of fringelets. Also included in this chapter are implementation details, pictures of fringelet-encoded images, and an analysis of fringelet performance.

6.1 Fringelet Generation

The essence of fringelet encoding is to construct a fringelet that possesses the desired hogel spectrum - or a spectrum that is closely related - using only N_h/CR samples. The driving concept was to make a fringelet look more like a hogel fringe so that decoding would be simpler and more rapid. The width of each fringelet is only a fraction of the width of the hogel for which it is encoded. Their relative widths determine the compression ratio (CR). For a hogel of width N_h=1024 samples, the fringelet width is 64 samples for a CR=16.

The fringelet for a particular hogel is computed from a hogel vector but using a special set of basis fringes. Each basis fringe has a width of N_h/CR samples, and is specially computed to contain spectral energy in a specific region. (See Appendix C.) Fringelet generation (direct-encoding) converts a hogel vector into a fringelet using the same type of superposition used to convert hogel vectors into hogels. The resulting fringelet has a spectrum with the desired amount of energy in each discretized region of the spectrum, centered at intervals of $\Delta_f = p\mathrm{BW}(\mathrm{CR}/N_h)$, where BW is the spectral bandwidth used (up to 0.5 cycles/sample). The important difference is that the fringelet has only N_h/CR samples. But a signal with only N_h/CR samples can have only N_h/CR distinguishable spatial frequencies. It is important to use basis fringes that leave some empty spaces in the spectral regions in between non-zero regions. This allows for energy compaction, i.e., bandwidth compression by eliminating useless information symbols. The fringelet was originally conceived as a “sparse-spectrum” hogel, i.e., a truncated piece of a hogel generated using basis fringes that would give the hogel a sparse spectrum. If such a hogel is truncated to only N_h/CR samples, the separate contributions in each region of the spectrum broaden but remain independent for reasonable values of CR.

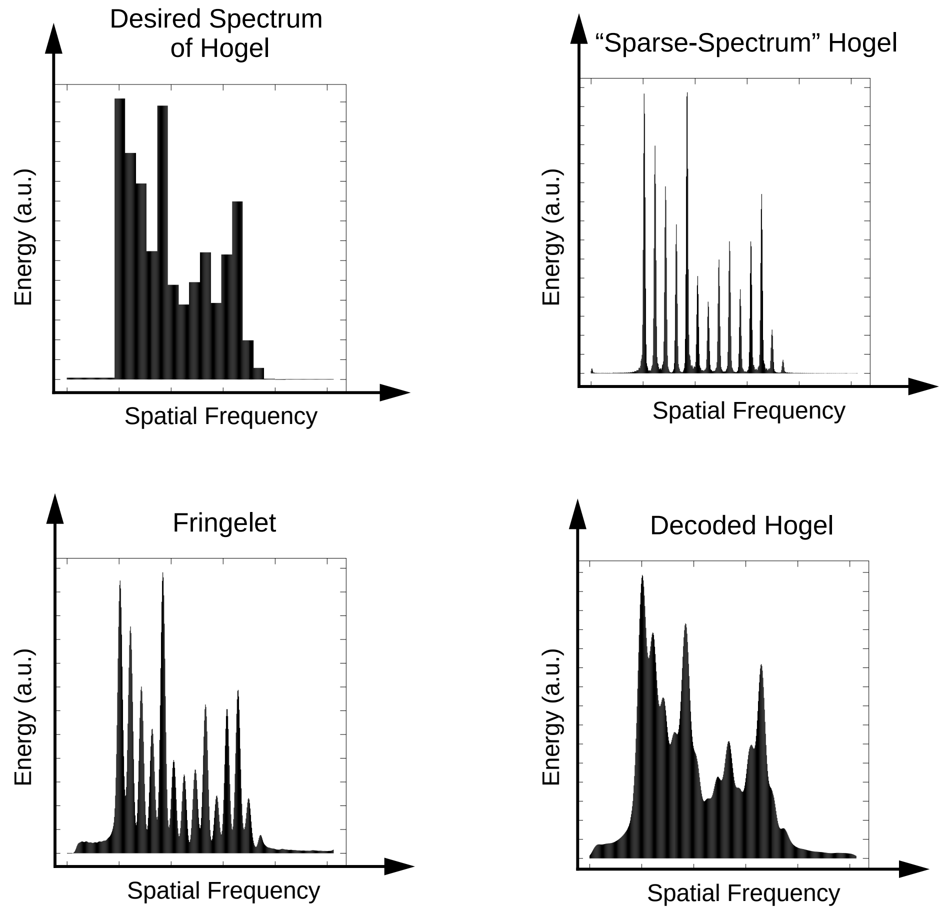
The fringelet spectrum is related to the desired spectrum as shown in the figure on page 107. A hogel vector represents a piece-wise continuous spectrum, with the amount of power in each region of the spectrum being proportional to the components of the computed hogel vector. The special set of basis functions are generated (using simulated annealing) to occupy a more sparse spectrum. The gaps in the spectrum allow for bandwidth compression without loss of spectral information. The spectrum of the fringelet still contains the same relative powers in each region of the spectrum.

6.2 Fringelet Decoding

The goal of fringelet decoding is to use a fringelet of width N_h/CR to create a hogel of width N_h - a usable fringe - that possesses the desired spectrum. This desired spectrum is encoded in the fringelet, the main difference being that some space exists between adjacent spectral regions. The decoding process must broaden the spectrum of each of the N_h/CR regions to produce a continuous spectrum. This is equivalent to the low-passing process required by the sampling theory to recover the sampled spectrum. The display process, including the diffraction of light, contributes to this desired spectral blur. The decoding process must also expand the N_h/CR-sample fringelet into a usable N_h-sample hogel. Fringelet decoding must perform the spectral broadening and the expansion without using complicated and time-consuming mathematical processing. Fringelet decoding is engineered to solve both of these problems.

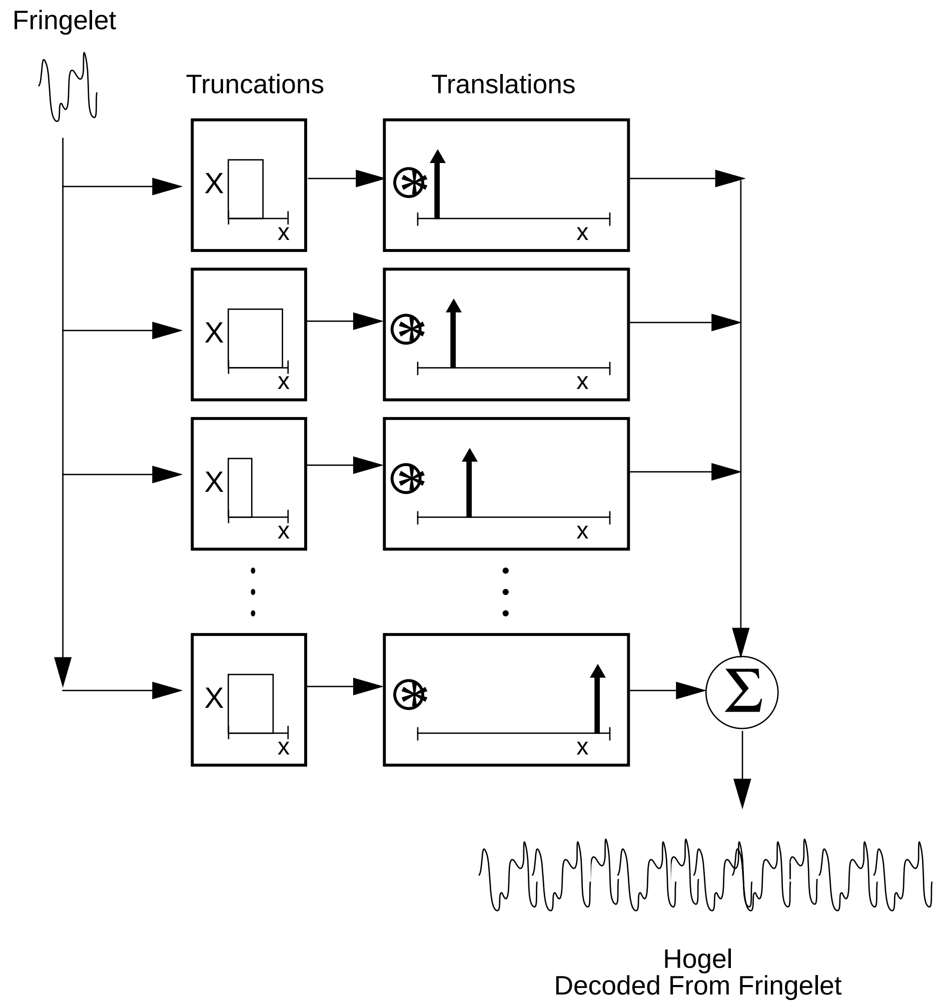
The process of fringelet decoding is illustrated on page 108. Replicas of the fringelet are truncated and translated by convolution with a series of stochastically spaced impulses. This analytical model looks complicated, but because the impulses have unity amplitude, there is no need for multiplication. In fact, if the replicated truncated fringelets do not overlap, then there is no need for any mathematical calculations at all! The only operations are the copying of fringelet values to hogel sample locations.

Fringelet and Hogel Spectra



This figure illustrates the spectra involved in fringelet decoding. At left is the desired spectrum. This spectrum was encoded in the fringelet, and subsequently decoded to produce a hogel. (The "sparse-spectrum hogel" is not part of the encoding-decoding progression, but is included for illustration.) The decoded spectrum contains the correct amount of spectral energy in each region. Notice that the decoded spectrum is continuous, whereas the desired spectrum (the information contained in the hogel vector used to compute the fringelet) is piecewise constant. A continuous spectrum is more desirable, since the diffracted wavefront will not have jumps that can lead to image artifacts. Thus, fringelet decoding actually improves the quality of the holographic information by spreading the spectral energy to produce a continuous spectrum that more accurately diffracts light to generate the intended wavefront.

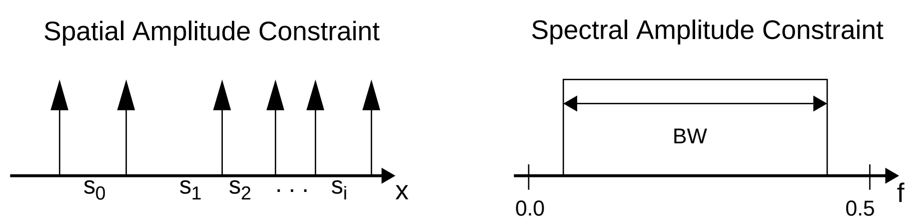
Fringelet Decoding: Block Schematic



As shown in this figure, fringelet decoding is performed by convolving a truncated replica of the fringelet with a series of stochastically spaced impulses of unity height. Randomly truncating (windowing) a fringelet has the desirable effect of spreading the spectrum in each of the regions. Convolution with a series of impulses has the desirable effect of expanding the width of the fringelet to match the width of the hogel.

The most important ingredient in fringelet decoding is the statistics of the truncations and replications. The truncations are relatively simple, giving rise to the correct spectral broadening for any random set of truncations. The mean value of the sequence of truncation widths determines the amount of spectral broadening. For simplicity, the truncation widths are set to match the width between each impulse in the convolution impulse sequence. This means that each replicated fringelet is truncated by precisely the same amount by which it is to be translated. This simplifies the decoding algorithm.

The statistical properties of the convolution impulse sequence are crucial to maintaining image fidelity. The impulses must have a spectrum that is as flat as possible. Recall that the convolution of the impulse sequence with the fringelet in the spatial domain acts to multiply the spectrum of the fringelet by the spectrum of impulse sequence. The spectrum of the impulses must be uniform to recover the desired spectrum. If the spectrum of the impulses has gaps or sharp peaks, then the spectrum of the hogel decoded from the fringelet will have noise and lead to image artifacts. To create an impulse sequence that satisfies all of these constraints, the simulated annealing algorithm (described in Appendix C) was adapted to this purpose. The constraints on the spatial and spectral characteristics of the impulse sequence are here illustrated and summarized:

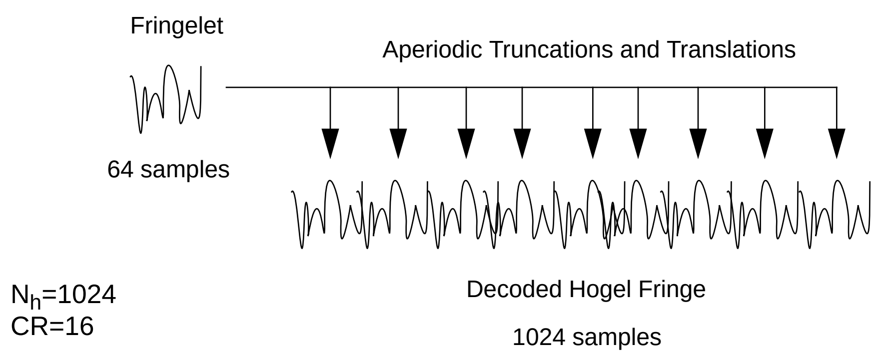


- Spatial amplitude: unity amplitude impulses (zero in between).

- Spatial amplitude: The impulses must occur at random (not periodic) intervals of s_i.
- Spatial phase: constant (zero).
- Spectral amplitude: impulse sequence must have a uniform spectrum across the spectral region of interest, and should be zero elsewhere.
- Spectral phase: unconstrained.

The starting point to the simulated annealing algorithm was a randomly distributed impulse with an average spacing that produced the desired spectral spreading. For each iteration, an impulse was chosen from the sequence at random and moved either forward or backward. This modification was either kept or rejected based on the probability function described in Appendix C. The desired impulse sequence converged after about 10000 iterations. Although this was a slow process (over 15 minutes), it only needed to be performed once for a given set of parameters. A separate replication sequence was computed in this way for all useful combinations of hogel width N_h and compression ratio CR. The separations s_i between the impulses is the amount by which the fringelet will be truncated and translated to decode it into a hogel. Once computed, this replication sequence was built into the decoding algorithm.

To help to visualize the actual fringelet decoding process, consider the example of a hogel width of N_h=1024, a compression ratio of CR=16, and a fringelet width of N_h/CR=64. To begin fringelet decoding, the 64-byte fringelet was copied into the first 64 hogel samples. (The replication sequence s_i was also constrained to make the first impulse spacing be equal to N_h/CR.) Next, the fringelet is replicated and truncated by s_1, the next value in the replication sequence. This truncated fringelet is then translated to fill hogel samples 64 through 64+s_1-1. Next the fringelet is replicated and truncated by s_2 and translated to fill hogel samples 64+s_1 to 64+s_1+s_2-1. This recursive aperiodic truncation and translation continues until the full hogel width of N_h=1024 is filled, as illustrated in the figure below. Each fringelet is subsequently decoded in this same way.



6.3 Implementation

Fringelet encoding and decoding were performed using the same computational platform as hogel-vector encoding. The first computational step was to generate the fringelet array on the Onyx workstation using the precomputed set of special basis fringes. This array of fringelets, with size (36 MB)/CR, was downloaded over the SCSI link to the Cheops P2 board. The i960 microprocessor on the P2 performed the fringelet decoding. The hogel array decoded from the fringelet array was then transferred to the Cheops output cards for display.

In the fringelet decoding step, the aperiodic truncation and translated replica of the fringelet is equivalent to taking each sample of the fringelet and copying it to several hogel sample locations. Each fringelet sample maps to a set of hogel samples, and each mapping is mutually exclusive and fully exhaustive. Therefore, this entire mapping process was stored in an indirection table on the P2. This indirection table was N_h entries wide, and each location contained the appropriate fringelet sample number.

Let the indirection table be INDtable[i], where i ranges from 0 to N_h-1. The following pseudocode describes the fringelet decoding algorithm using the indirection table.

```
For each Fringelet: {
    For each Hogel sample: {
        Hogel[i] = Fringelet[ ( INDtable[i] ) ]
    }
    Load Hogel into appropriate output card location.
}
```

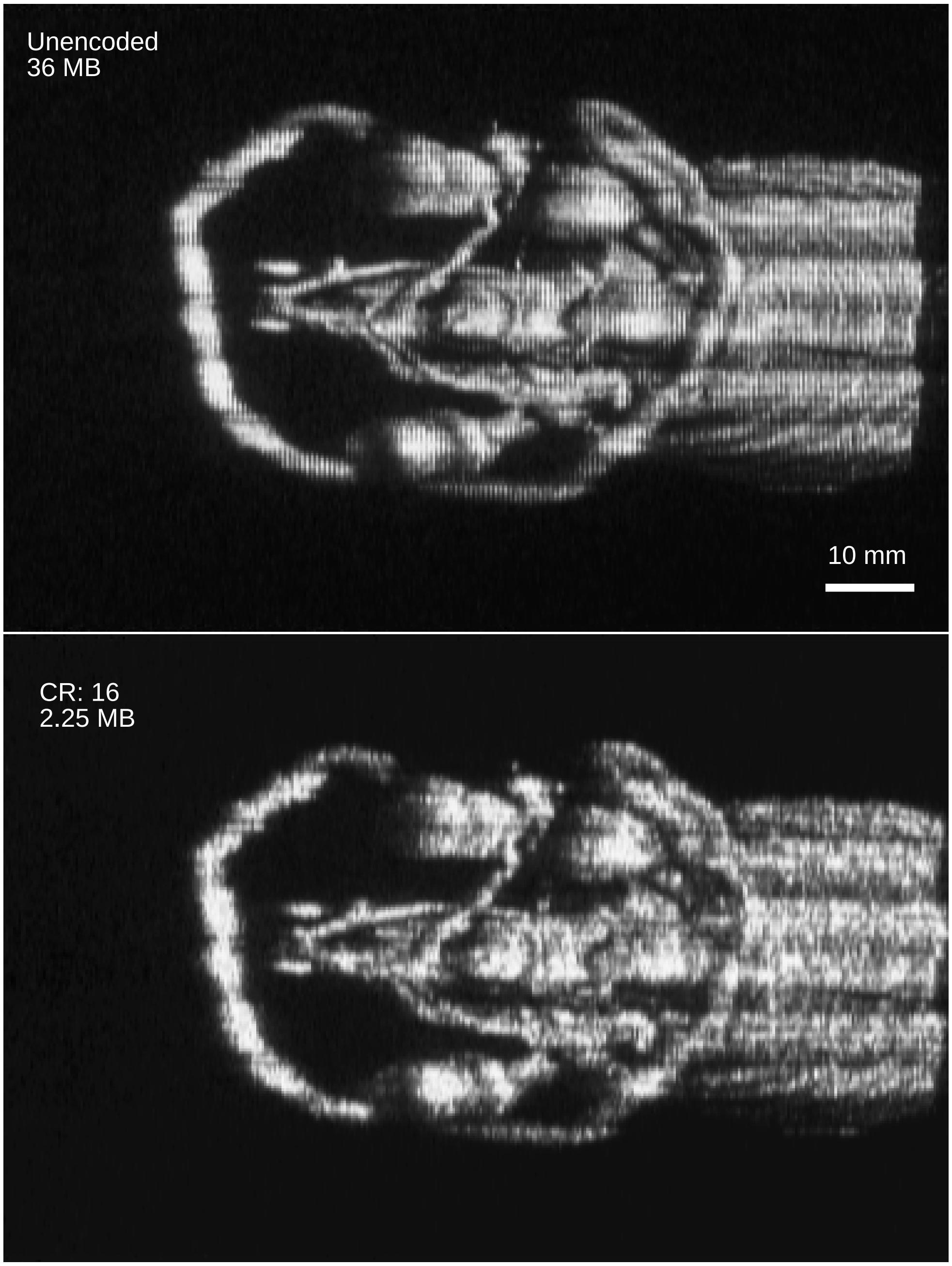
The only operations are (1) fetching the value from the INDtable, (2) using that value to fetch the appropriate fringelet sample, and (3) copying the fringelet sample value into the indexed hogel sample location. Each decoded hogel value is decoded using no math and only a few memory operations. As discussed later, the speed of fringelet decoding is very fast due to this extreme simplicity.

The implementation of fringelet encoding and decoding included two esoteric details. First, the lowest spatial frequency contained in any fringelet was nonzero, limited to allow the fringelet to contain at least one period of this frequency. (For typical fringelet parameters, only 3 per cent of the available bandwidth was wasted as a result.) Second, each fringelet basis fringe was constrained such that its average value was equal to the average between the minimum and maximum values. This eliminated a significant source of noise generated in the decoding step.

6.4 Image Generation

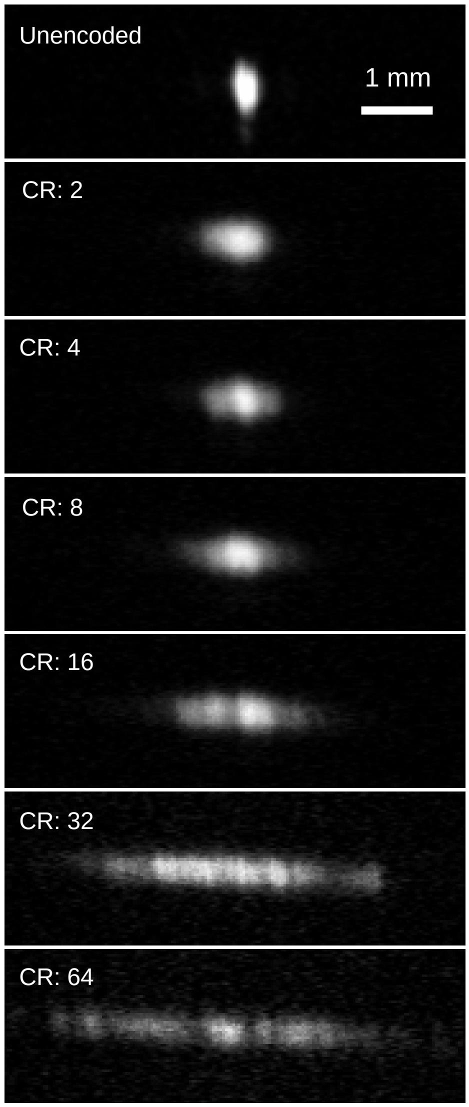
To illustrate the effect of fringelet encoding on holographic image quality, several images were digitally photographed, both close-up and full image. Several examples were documented, using images at different depths, and using different values of hogel width N_h and compression ratio CR. The digitally photographed images were gathered together and shown on pages 113-116. The profile of each focused point was also acquired by sampling a cross-section of the central region of each data set. The illustrations on page 116 show an example of a cluster of imaged points, photographed in the same way as the single points, showing the interaction between neighboring points on a hololine.

Rocket Engine Fuel Intake



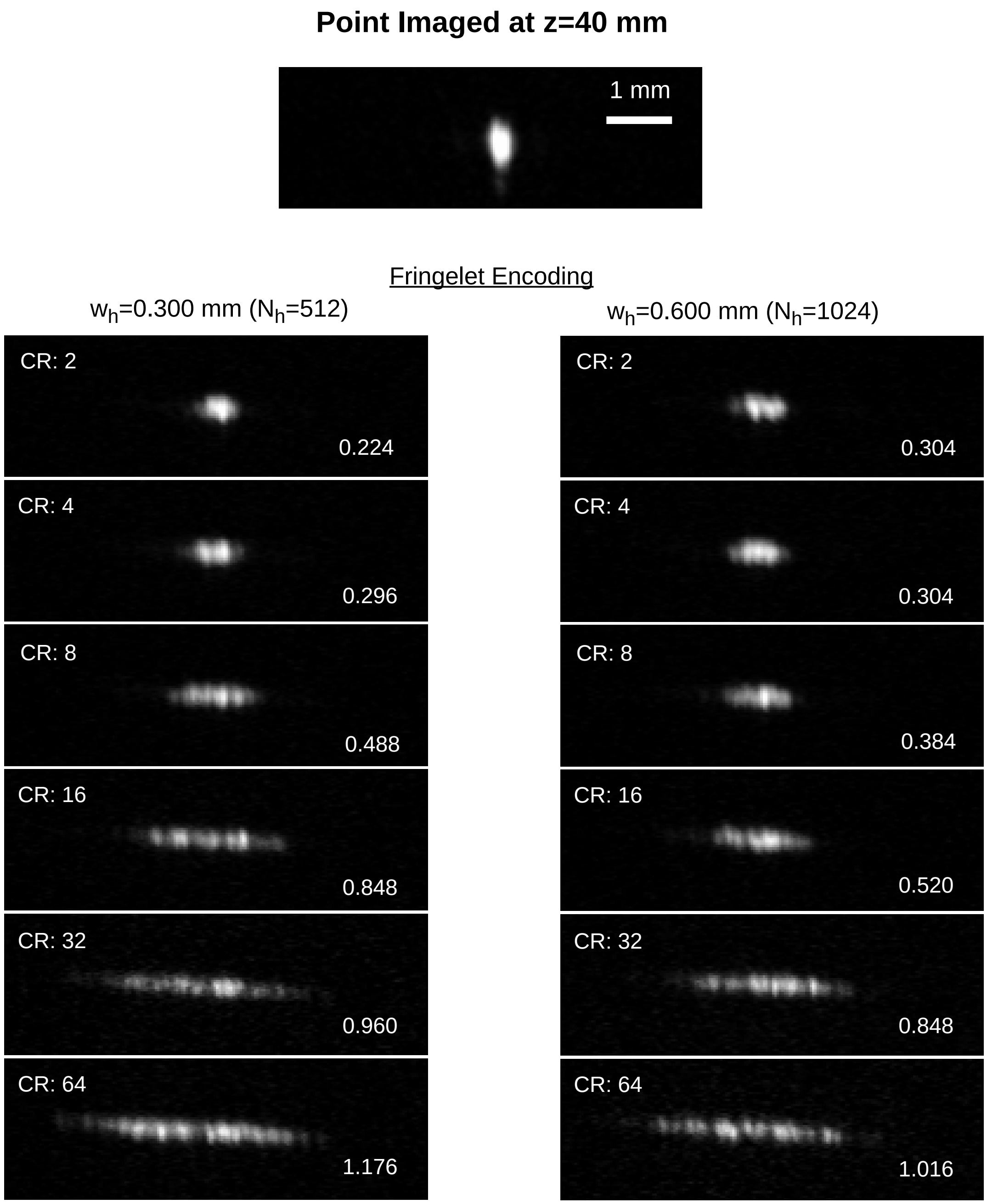
Each of these illustrations is a photograph of a 3-D image of the fuel intake system of a Space Shuttle rocket engine. The unencoded image (top) shows the discrete points of light that compose the image. This sharpness is lost when the image is fringelet compressed with a CR=16 (bottom). Nevertheless, the blur added to the image does not severely degrade image fidelity.

Fringelet Encoding:
Point imaged at z=80 mm, w_h=0.600 mm (N_h=1024)



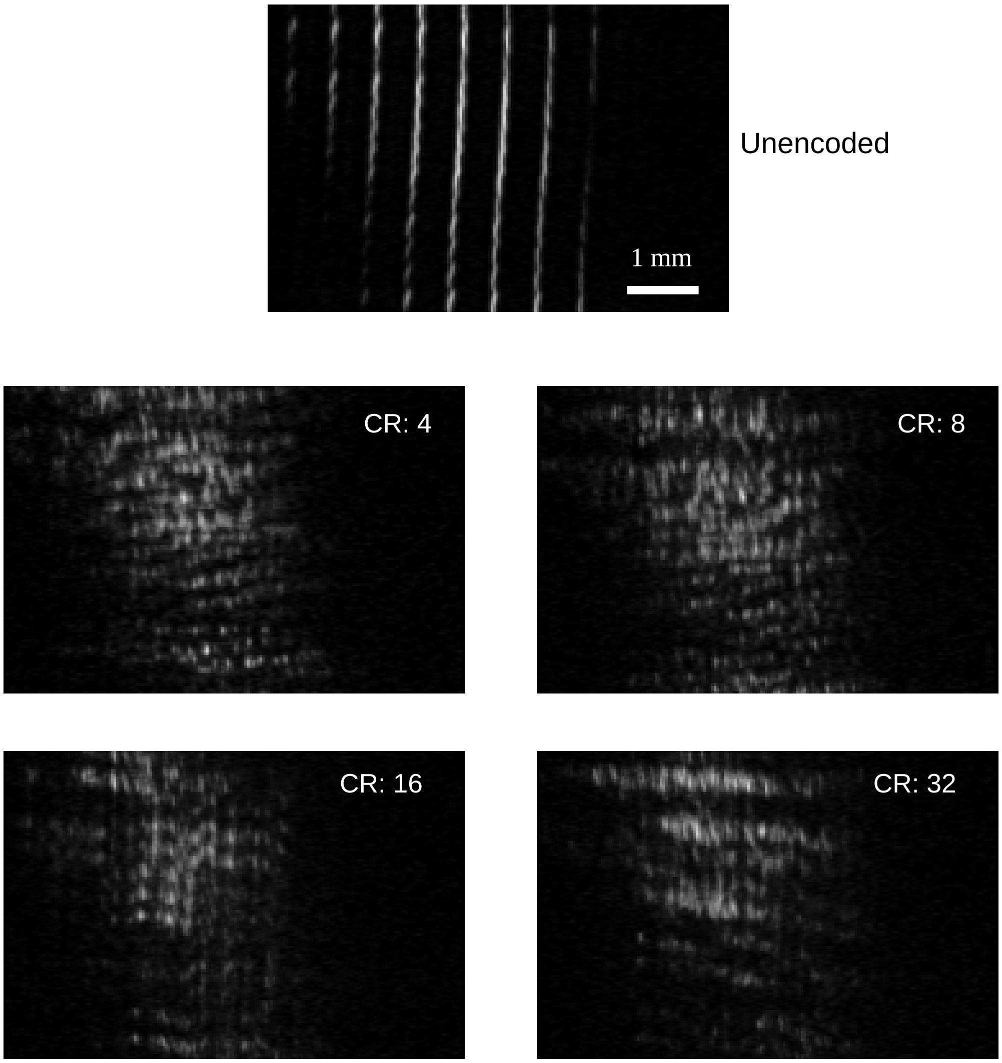
Eff. width: 0.144 mm
Eff. width: 0.368 mm
Eff. width: 0.376 mm
Eff. width: 0.456 mm
Eff. width: 0.712 mm
Eff. width:1.288 mm
Eff. width: 2.200 mm

This figure shows a series of cross-sections of a point imaged at z=80 mm for a range of compression ratios. This is the worst-case situation: 80 mm is the maximum depth used in this display. The point blurs (horizontally) as CR increases. For CR of 8 or lower, the point is still relatively sharp. For compression of CR=16, the point begins to blur to a width that is easily seen by the human viewer. Hogel width for all of these images was w_h=0.600 mm, or N_h=1024 samples.



This figure shows a series of point images focused at z=40 mm. To illustrate the effect on point spread of different hogel widths, the point is shown for hogel widths of 0.300 mm and 0.600 mm over the range of compression ratios. The measured effective widths (in mm) are listed in the lower right corner of each cross-section. Although the two values of w_h are more similar than in the case of the z=80 mm point, again the choice of w_h=0.600 mm gives less point spread at high values of CR.

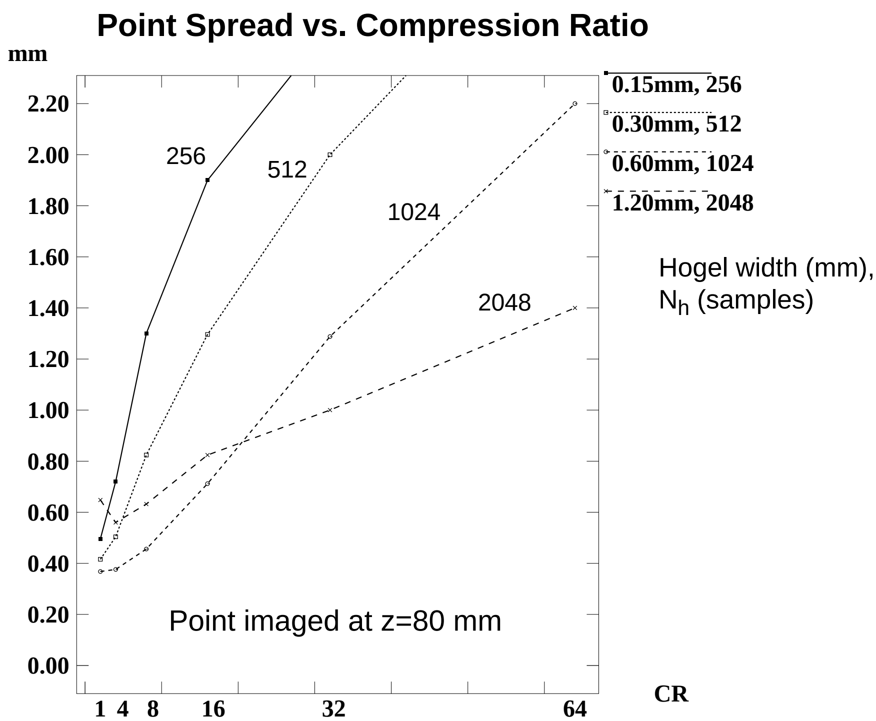
Fringelet Encoded Image: Close-up

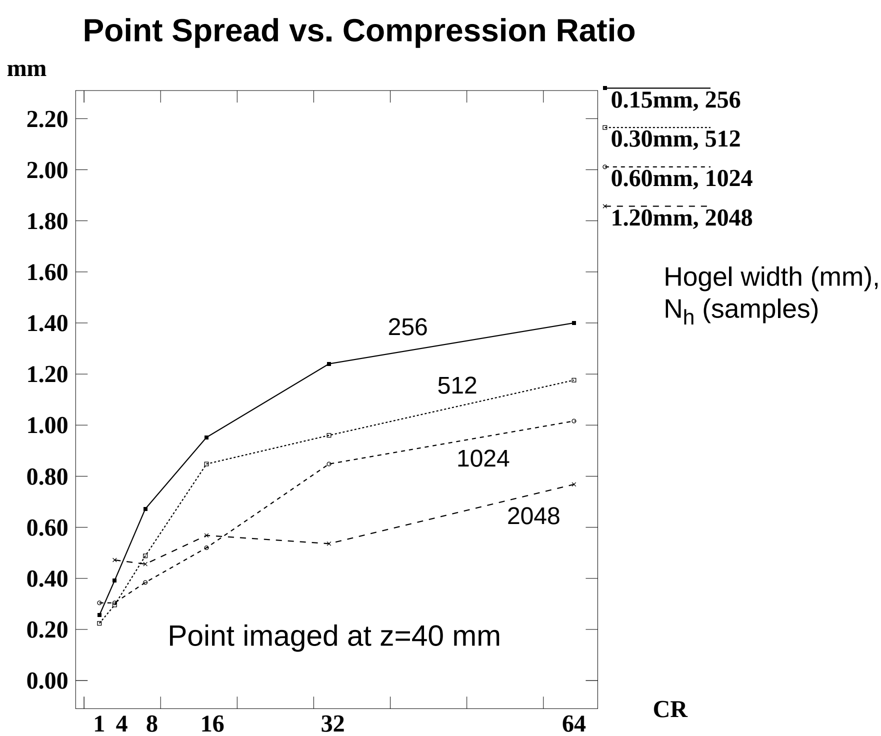


This figure shows close-ups of a full image (of a rocket engine) computed using fringelet encoding. The CCD array was placed in a region at z=40 mm containing a surface (represented by an array of imaged points) as well as regions of black (no image elements). In the unencoded image, the array of imaged points is clearly visible. For the fringelet encoded images (w_h=0.600 mm, N_h=1024), the blur of each points has the desirable effect of joining the discrete image points together to form a continuous surface.

To measure the image resolution achievable using fringelet encoding, a series of point images was acquired and their point spread calculated. For the worst-case point at z=80 mm, fringelet encoding produces point spread that is unacceptable for high values of CR. The graph at the top of page 118 plots the measured point spread for a range of compression ratios and hogel widths. For CR=16, a hogel width of w_h=0.600 mm (N_h=1024 samples) produces reasonable results, even in this worst-case image depth.

Fringelet encoding performs much better for points that lie within the more commonly used regions of the image volume. The lower graph on page 118 shows the measured point spread for a point imaged at z=40 mm. Point spread is reduced by a factor of 2 as compared to the point imaged at z=80 mm. This indicates that the primary source of the blur is spectral, which causes a spreading that increases linearly with diffracted distance. As for the worst-case image point, the best results for CR=16 were for N_h=1024, w_h=0.600 mm.


Point Spread vs. Compression Ratio
mm
2.20
2.00
1.80
1.60
1.40
1.20
1.00
0.80
0.60
0.40
0.20
0.00
256
512
1024
2048
Point imaged at z=80 mm
1 4 8 16 32 64
CR
0.15mm, 256
0.30mm, 512
0.60mm, 1024
1.20mm, 2048
Hogel width (mm),
N_h (samples)


Point Spread vs. Compression Ratio
mm
2.20
2.00
1.80
1.60
1.40
1.20
1.00
0.80
0.60
0.40
0.20
0.00
256
512
1024
2048
Point imaged at z=40 mm
1 4 8 16 32 64
CR
0.15mm, 256
0.30mm, 512
0.60mm, 1024
1.20mm, 2048
Hogel width (mm),
N_h (samples)

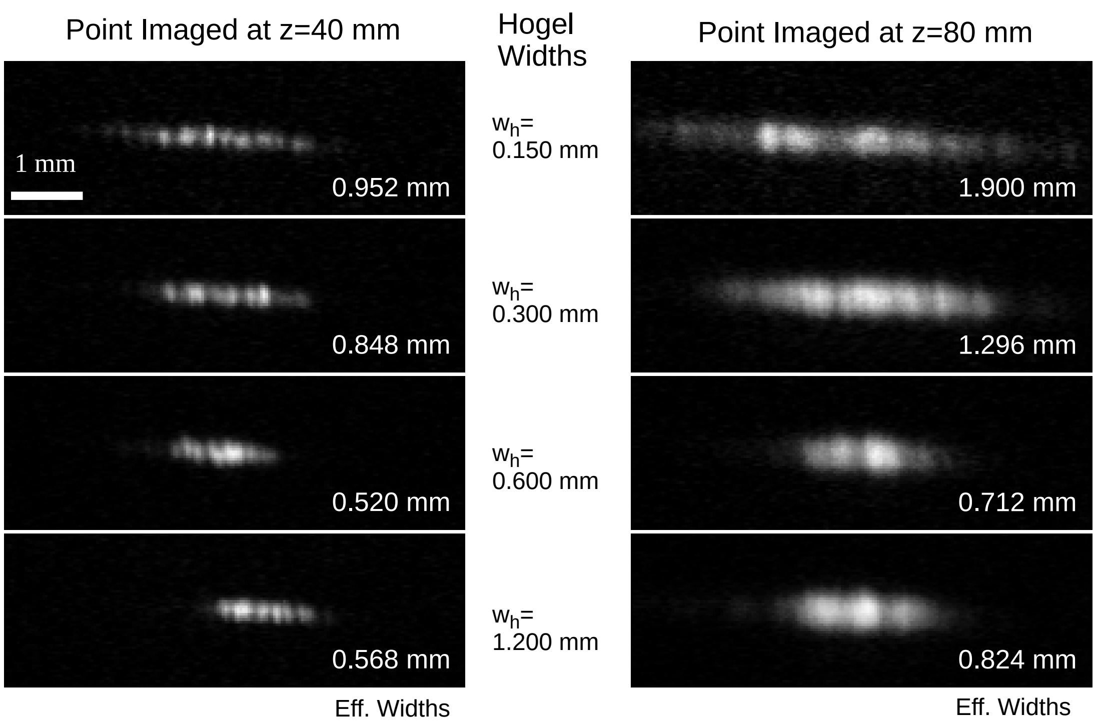
6.5 Discussion of Point Spread

Fringelet encoding adds more point spread than does hogel-vector encoding. The primary cause is spectral sampling blur and crosstalk. The number of spectral regions to be encoded in a fringelet is N_h/CR, the same number of samples in the fringelet. However, the spectrum of each component becomes broad for small values of N_h/CR. The resulting spectral cross-talk is the cause of the additional blur. Because of this crosstalk, fringelets do not allow for completely independent control over separate spectral regions. However, for reasonable values of CR, this crosstalk adds only a small amount of image noise. Notice in the graphs on page 118 that the point spread decreases dramatically as CR decreases or as hogel width (N_h) increases. For these larger values of N_h/CR, cross-talk is limited to nearest neighbors only. For smaller values, i.e., fewer spectral samples, cross-talk begins to affect broader ranges of the spectrum. This cross-talk essentially places a maximum limit on the entropy per symbol in the holographic communication system.

As was the case in hogel-vector encoding, choice of hogel width is important to encoding performance. The figure below shows the effect of different hogel widths, w_h. A point is imaged at z=40 mm and at z=80 mm using fringelet encoding for a range of hogel widths. The measured effective widths (in mm) are listed in the lower right corner of each. As previously noted, the selection of w_h=0.600 mm (N_h=1024) provides optimal performance. Notice also in this figure that the shape of the imaged point has some structure. These variations are caused by imperfections in the fringelet decoding process. the impulse sequence does not have a perfectly smooth spectrum. Each focused point is essentially multiplied by an envelope function that is a representation of the actual spectrum of the impulse sequence.

6.6 Speed

The use of fringelet encoding increased holographic fringe computation speed by a factor of 2.8 without requiring specialized hardware. Decoding a fringelet involves no



Compression Ratio = 16

mathematical calculations. Using the indirection table described earlier, only memory access and byte replication is necessary. It can be performed as quickly as data can be transferred within the P2 board. Total fringe calculation time using fringelet encoding was typically 9 seconds, distributed as follows: 5 seconds for the fringelet computation (direct encoding), 2 seconds for the SCSI download, and 2 seconds for the decoding including the time to transfer to the VRAM. Transfer to the VRAM of the output cards used the Nile bus. Fringelet decoding accounted for 1.2 s and Nile transfer of the 36-MB decoded fringe pattern accounted for 0.8 s. These times are for fringelet decoding implemented on the i960 microprocessor on the Cheops P2 board. Fringelet decoding is fast even without the use of specialized hardware.

Note that the decoding time for generating 36-MB fringes was 1.2 s compared to 20 s for hogel-vector decoding implemented on special hardware (the Splotch Engine). To more accurately compare the two decoding schemes, each was implemented on the Onyx using only standard serial C code. On the same computational platform, fringelet decoding is consistently over 100 times faster than hogel-vector decoding. The trade-off is that the first computational step - generation of the fringelets - requires more time than the generation of hogel vectors. (The fringelets are computed from the hogel-vector array, so some additional computation is always required to generate the fringelets.) Nevertheless, total computation time is down by a factor of more than 2.0 compared with hogel-vector encoding.

The following listing summarizes the timings for fringelet encoding and decoding. The first two numbers indicate times for direct-encoding and for decoding. These times sum to total computing time (excluding transfer time).

- Onyx → Cheops P2 → VRAM: 5 s + 2 s =7 s
- SCSI transfer time: add 2 s

The goal of fringelet encoding has been achieved: the decoding algorithm is so simple that it can produce a 36-MB fringe in under two seconds - fast enough for interactive computation. A system designed to convert hogel-vectors to fringelets and fringelets to hogels can be implemented on specialized hardware to reduce speeds to interactive rate. Chapter 8 discusses the possibilities of implementing fringelet decoding in digital hardware, in the analog RF electronics, or optically in a specially designed holographic “fringelet” display system. First, however, Chapter 7 further compares and contrasts the performance of hogel-vector versus fringelet encoding, to determine which is more appropriate for a given imaging requirement.