A Difference Engine for Image Reconstruction [1)]

E.H. Blake

Department of Computer Science, University of Cape Town
Rondebosch, 7700 South Africa
Email: edwin@cs.uct.ac.za

A.A.M. Kuijk

Department of Interactive systems, CWI
Kruislaan 413, 1098 SJ Amsterdam, The Netherlands
Email: fons@cwi.nl

ABSTRACT

A difference engine is described that is designed to be used as low level component of a raster graphics architecture. The speed of the system (11ns per operation) is achieved by the use of custom VLSI components for the most primitive operations. This permits the video rate reconstruction of images and other signals compressed by encoding them on various polynomial bases. The paper describes a feasibility study for its use for image reconstruction. The study shows that the system can be applied to the decompression of spline wavelet encoded images.

1. Introduction

A reappraisal of the three-dimensional (3-D) interactive raster graphics pipeline has resulted in an experimental architecture for a graphics workstation which has been presented in previous workshops on graphics hardware [19-21, 24]. Some of the novel uses of parts of the hardware were not foreseen when the research project was initiated. In this paper we shall explore one of the unanticipated spin-offs from the project.

The principal features of the design for the raster graphics architecture were:

- Emphasis on real-time interactive shaded 3-D graphics.
- Object space methods rather than image space methods are used where possible.
- The use of a frame buffer is avoided.
- Custom VLSI is used only at the lowest, most primitive, levels where commercial products are unlikely to suffice in the near term.

It were these design decisions that lead to a number of interesting consequences that have made parts of the architecture eminently suited to a far wider range of problems in computer graphics and image processing. One of the design phases concentrated on extracting the lowest common denominator of primitive operations for synthesizing pixels —a language for manipulating related pixels. This vocabulary is sufficiently general to be used for expressing other facts about images.

The custom VLSI development that was a major part of the project produced what is essentially a very fast *Difference Engine* (to borrow a term from the 19th century history of computation). This Difference Engine is the element of the graphics architecture that actually executes the primitive operations for synthesizing the pixels. It can compute forward differences

[1)] Modified version of [1]

in parallel over the whole width of a typical image, taking about 11ns per operation (90 Mhz clock) independently of the length of the forward difference spans.

In the next section we present a very brief overview of the architecture in terms of its original design for producing real-time interactive raster graphics. In Section 3 we show how the same low-level architecture has applications in the reconstruction of compressed signals, and we present the results of an initial feasibility study. In Section 4 we introduce wavelets as a new generalized and promising approach to studying compression and decompression schemes for sounds and images. Section 5 is on ongoing development. In conclusion (Section 6) we summarize our results and point to some very promising areas for further research into applications of the hardware under discussion.

2. The Difference Engine

In the past four years the design of the raster graphics architecture has been completed [21], the critical components have been made [22] and functions simulated in detail [19]. The functional elements of the raster graphics architecture are shown in Figure 1. The array of *X* processors and the *Y* processors together form the display controller.

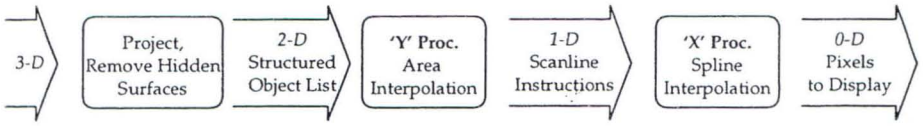

Figure 1: Functional Elements of the Display Architecture. *The Y– and X–processors together form the display controller of the graphics architecture. The arrows show how 3-D data are converted to a stream of pixels. Data structures are indicated by arrows and of these only the 3-D Models and the 2-D Structured Object List are stored. All other data are generated on the fly. The Structured Object List is updated at the animation or interaction rate, 12—24 times per second, and it is read at the video frame rate, 50—60 cycles. The Scanline Instructions are produced at the video line rate and clocked into the X processors at the video pixel generation rate.*

The *X* processor array (which deals with the scanlines of an image) is actually a Difference Engine. That is, it can do forward difference calculations at high speed. Any order of forward difference can be done, the limiting factor is the accumulation of errors during addition and (in real-time applications) the extra cycle time taken by each higher order. The architecture is tailored to second order forward differences.

The Difference Engine is a systolic array with a dedicated processor for each pixel in a scanline [2)]. Additions (and input) are done to a precision of 36 bits and the top 12 bits are output. So we basically have data values of 12 bits — say 10 bits magnitude, 1 sign bit and 1 bit to detect common overflow situations — and 24 precision bits. For quadratic interpolation (second order forward differences) spans of 2^{12} bits or 4096 pixels wide can be interpolated before the error becomes noticeable. This is also the maximum addressable pixel and span width allowed for input. If we use just 9 data bits of the output (i.e., 8 bit values + sign bit) then cubic interpolation for spans of 512 pixels can be done accurately. Naturally longer spans can always be done by splitting them into shorter ones which can be done correctly.

[2)] Actually one can have fewer processors provided the number of output pixels are a multiple of the number of processors. The processors are then reused for the remaining parts of a scanline, but the normal situation is to have one processor per pixel in a scanline.

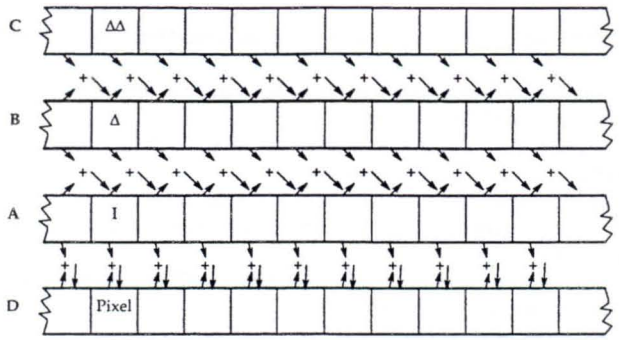

Figure 2: Interpolation on the Difference Engine. *The Difference Engine is implemented as a systolic array of processors: one for each output pixel. For quadratic interpolation the second differences remain constant and are propagated to all processors within an active span of pixels (Register C). The first differences (Register B) are changed by the second differences at each step and the results added to the intensity values (Register A). The results of the interpolation step are added to an accumulator (Register D) so that multiple interpolation spans may overlap to produce the final pixel value. For higher order differences the registers are reused.*

The bias towards quadratic interpolation is also seen in the fact that there are 3 internal registers with which the value I (''intensity''), first difference δI and second difference δδI can be set at every pixel location. The operation of the Difference Engine is illustrated in Figure 2. The interpolated intensities are calculated in register A and the contents of A is added to an output accumulator which can contain the results of a number of previous interpolation spans. The contents of the accumulator is read out (for display) on receipt of a ''refresh'' instruction. Other instructions are listed in Table 1.

Higher order differences are done by reusing registers. In fact the registers are not needed to perform forward differences but only so that differences may be set ahead of time by means of the 'set' instructions (see Table 1). It is a way of changing just certain (usually the highest) differences within an interpolation that is continuous in the lower orders.

To conclude, this *X* processor which was developed as a very specialized pixel generator is really very general; it is a *Difference Engine* with:

- Any order forward differences.
- 36-bit numerical accuracy and an 11ns cycle time.
- Any spline interpolation, with constant cost independent of span length.

3. Reconstruction of Compressed Signals on a Difference Engine

Clearly the Difference Engine can interpolate any spline (polynomial) curve. Thus any signal that is expressed in terms of a *spline basis* can be reconstructed. Not only that, the architecture with its accumulator allows one to sum over incrementally generated output so that the splines can be summed over different scales to produce the final image to any required accuracy.

3.1. Simple Compression

Obviously run length encoding is handled by a single instruction ('eval0', see Table 1). One could also extend the concept to using linearly increasing runs, although the benefit is doubtful.

operation	description	cycles
acc_mode	Accumulate mode: if enabled negative intensities are not added to accumulator.	1
dis(x,dx)	Disable accumulation of intensities from pixel 'x' for 'dx' pixels (cleared after next 'eval*' command).	1
eval0(x,dx,i)	Set pixel (i.e., accumulator) from 'x' for span of 'dx' directly, disable further additions until next refresh.	1
eval1(x,dx,i)	Add i to accumulator for span from 'x' for 'dx' pixels.	3 [a]
eval2(x,dx,i,di)	First order forward difference, starting at pixel 'x' with value 'i' and increment 'di', for 'dx' pixels.	4 [a]
eval3(x,dx,i,di,ddi)	Second order forward difference —like 'eval2' except now 'di' is also changed by 'ddi' at each step.	5 [a]
eval4(x,dx,i,di,ddi,dddi)	Third order forward difference, like 'eval3' *mutatis mutandis*.	6 [a]
eval n	Higher, n-1, order forward differences.	n+2 [a]
nop	No operation.	1
refresh	Output accumulator value and clear everything.	2
setddi(x,dx)	Set (i.e, override) second difference [b] at points 'x', for a span of 'dx' pixels in the middle of the next 'eval' command.	2
setdi(x,dx,i)	Like 'setddi' only it affects the lower forward difference.	2
seti(x,dx,i)	Like 'setdi' except that this creates a span of intensities.	2
setpddi(x,dx)	Set (i.e, override) second difference [b] at points 'x', 'x+dx', 'x+2dx', ... in the middle of the next 'eval' command.	2
setpdi(x,dx,i)	Like 'setddi' only it affects the lower forward difference.	2
setpi(x,dx,i)	Like 'setdi' except that this creates a pattern of intensities.	2

Table 1: *X* Processor Instructions and Their Costs in Cycles. *Note: The costs mentioned above are incurred whether a span is 1 pixel long or covers the whole width of the scanline.*

[a] *The cost of this operation can be reduced by 1 cycle in future versions.*
[b] *If there are higher order differences then this sets the highest order difference.*

The intended benefit of compression is to save both space and time, space is saved since images become smaller and the transmission times also decrease. The cost is of course having to perform the compression step first and then the decompression. In a number of situations the most time critical step is decompression, where images have to be viewed in (near) real-time. The Difference Engine can greatly assist with this as it can interpret the actual image coding instructions directly [3].

3.2. Expressing an Image in a Multi-Resolution Polynomial Basis

A simple experiment has been run to demonstrate how the Difference Engine can be used to reconstruct an image expressed in a multi-resolution polynomial basis. The interest of the demonstration is to show how reconstruction would work in practice —not (yet) to uncover a new compression technique. The image encoding method will be described below. The results of the encoding was a stream of *X* processor instructions that could be truncated at any desired resolution and consisted of full size descriptions of the image at each resolution level. A number of these are shown in Figure 4. These instructions were interpreted by a fast systolic array simulator to recreate images which were more or less blurred as desired (see Figure 4).

[3] Image compression steps are frequently followed by more traditional and well understood data compression algorithms, we do not discuss those here.

A Simple Encoding Algorithm

To illustrate the usefulness of the architecture introduced in Section 2, a multiresolution encoding of the image was needed. A simple algorithm was developed for test purposes which we believe illustrate the essential ideas while having a very low computational cost. It works only in 1-D along scanlines and requires no convolutions [4]. The image can have any size within the address range of the *X* processor.

Since compression was not the objective a very simple basis function was chosen: a parabola of the form $1-x^2$ defined on the domain $-1 \le x < 1$. This length is then scaled to spans of width 2^n, where $2 \le 2^n < 2 \times$(width of image-1), $n \in Z$, and translated in steps of 2^n where necessary to tile the whole scanline. Various small optimizations are possible to handle symmetries and very short spans. The operation of the algorithm on a single scanline is illustrated in Figure 3, the steps are as follows:

- The "compression" algorithm first finds the linear trend in a scanline and produces the instructions which would interpolate that linear term.
- The linear span is subtracted from the input image. This produces the difference image which is illustrated on the top left in Figure 4.
- Subsequently the first quadratic span whose length is a power of two and whose centre falls within the image width is chosen, i.e., span length = 2^n where 2^{n-1} describes a pixel inside the image, see Figure 3. The value of this pixel *alone* determines the height of the parabola (no convolutions or other complications).
- As with the linear term, the scaled values of the parabola are subtracted to produce a new difference image.
- The next span length is half the previous one and the procedure is repeated recursively. Note that the one pixel whose value was chosen is never changed by higher resolution (smaller) parabolae.

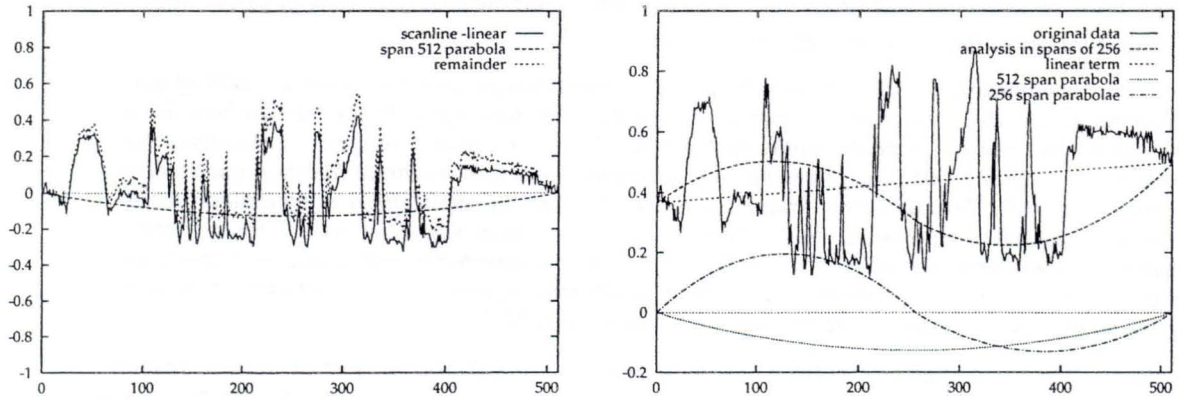

Figure 3: Encoding A Single Scanline. *The diagrams illustrate the situation in encoding a single scanline that contains the eyes from Figure 4. The diagram on the left is the situation where the first parabola is "fitted". This has a span of width 512, the parabola of width 1024 would have been needed if the image was just one pixel wider. The diagram on the right shows the results, and the basis functions, of "fitting" spans 512 and 256 pixels wide.*

[4] The astonishing thing is that this algorithm actually produces a useful image decomposition.

Results

The results of applying the procedure and then reconstructing the images on the *X* processor simulator are illustrated in Figure 4. It can be seen that it is possible to decode a low resolution image directly and that the natural way of reconstructing the image is to build up the output resolution level by resolution level. In a real-time application one could therefore have graceful degradation of performance since the display controller could truncate the image being processed after a certain number of instructions and one would still have a useful image. In some pyramid systems this is more of a problem since the higher levels of the pyramid (the lower resolutions) actually are smaller images and need to be expanded for display purposes.

The compression procedure outlined above could be turned into a true compression of the image by adopting a pyramid compression scheme as outlined in [3] but coding it explicitly in terms of the basis functions.

One feature of note in our simple coding scheme is that the actual edge information is retained far too long. This visually important information should really be sent early on with the low resolution image. Other more sophisticated compression schemes also suffer from this problem. Thus an important area of investigation is to find compression algorithms that send "important" information first, not merely low resolution information. One could also conjecture that such algorithms will be more efficient as lossy encoders of images destined for viewing by people (see also Sections 4.3, 5 and 6).

4. Wavelets

Wavelets are a "new" decomposition tool for analysing signals at multiple scales. Multi-resolution analysis has been around for a long time but recently there has been significant advances both in unifying old ideas in a single theory and in deriving new results. Indeed a recent introduction to the topic has referred to it as:

> *"...this comparative frenzy of research activity ... which has provided not only a wealth of new mathematical results, but also a common language and rallying call for researchers in a remarkably wide variety of fields: mathematicians working in harmonic analysis ...; mathematical physicists ...; digital signal processors because of connections with multirate filtering, quadrature mirror filters, and subband coding; image processors because of applications in pyramidal image representation and compression; researchers in computer vision who have used "scale-space" methods for some time; researchers in stochastic processes interested in self-similar processes, 1/f noise, and fractals; speech processors interested in efficient representation, event extraction, and mimicking the human auditory system. And the list goes on [13]."*

Wavelets or "the principles of multiresolution analysis" are still the subject of intense research. It is however clear that it shows great promise for image compression (see Section 4.3). Indeed this kind of analysis — as can be seen in the above quote — has quite a long pedigree in image processing dating back to the early eighties [3,4,35]. What we have demonstrated in the previous section is that if an image can be expressed in terms of a spline basis then it can be reconstructed on our image synthesis hardware. In the remainder of this section we would like to show by referring to the literature on multiresolution analysis, some of which is very recent, that this can be a very powerful technique.

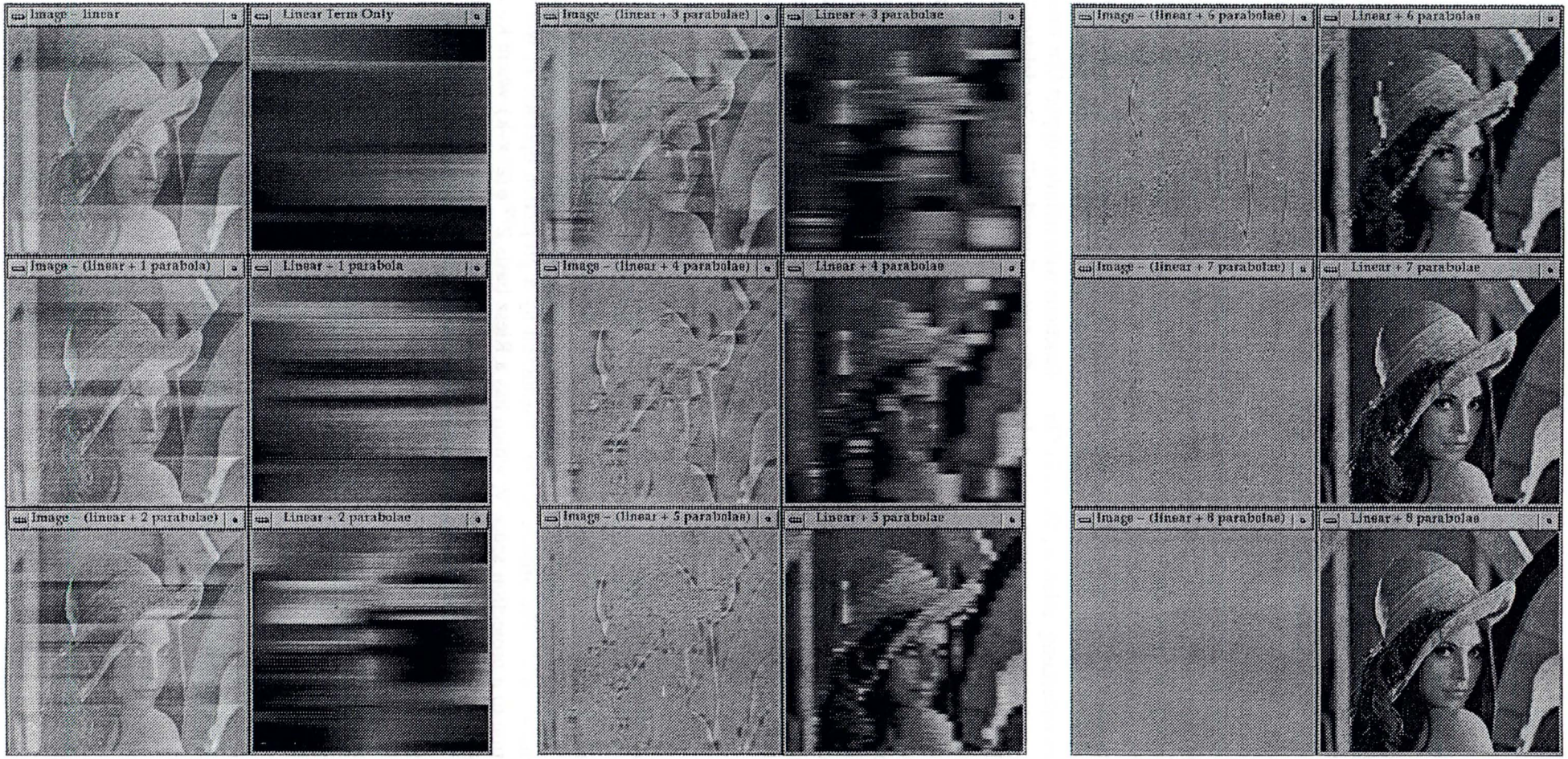

Figure 4: Image on Parabola Basis, Steps 0—8. *The images on the leftside of each block are the remainders after the encodings on the right have been subtracted. The leftside images have been scaled so that normalized pixels with the value of -1 are black and +1 are white. On the right 0 is shown as black and 1 as white, while negative values, which can exist in intermediate images, are suppressed. The right hand images were executed on a simulation of the difference engine.*

4.1. Wavelets and Spectral Analysis

This is not a tutorial on wavelets: we simply wish to show the applicability of the field to our area of application. If we incidentally motivate readers unfamiliar with wavelets to study the subject then one of the numerous introductions that now exist can be consulted (even in the popular literature, see [9]). Particularly appropriate for our area of interest are [5, 25], general introductions can be found in [10, 11, 30] while useful collections of recent papers can be found in [6, 12, 29] ([6] contains a extensive bibliography).

When the spectrum of a signal is to be obtained by Fourier analysis then all information about the signal, past and future, $-\infty$ to $+\infty$ in all dimensions, must be available. If the signal is altered in some small neighbourhood of a point then the whole Fourier spectrum is affected. Considered another way, if we know that a signal is exactly located as an impulse $\delta(t-t_0)$ at t_0 then its spectrum is spread out over the whole frequency domain as $e^{-it_0\omega}$. Alternatively if we have an exact frequency then the location of the pure sinusoid is indeterminate. Wavelets offer a general compromise in this uncertainty relation, instead of basis functions with infinite support like sines and cosines they use bases with compact support, hence the terms ''wavelets'' or ''ondelettes''. An analogy are the notes used to score music: they specify a particular tone at a specified time with various scales of duration.

An early example was the Gabor transform that offered the theoretically optimal combination of location in frequency and space for a fixed size window of uncertainty. Wavelets have a lower simultaneous accuracy in the two domains but have other advantages over early approaches. They automatically zoom in to a high frequency and locate it more accurately and they zoom out to a wide window to analyse low frequencies more accurately. Another characteristic of wavelets is that there is not a unique set, in fact there are infinitely many. We shall consider spline wavelets since they are useful for image compression and are suited to our decompression system.

4.2. Multiresolution Analysis

Wavelet bases are often defined from the data produced by a multiresolution analysis.

A signal $f(x)$ which is ''well behaved'' in a precise sense ($\in L^2$) can be approximated at various resolutions 2^j, $j \in Z$. The approximation is achieved by a projection operator P_j which projects the function to a projection space V_j which has a Riesz basis $2^{j/2}\phi(2^j x-k)$ where $k \in Z$ and $\phi(x) \in L^2$. In this way we obtain a nested sequence of subspaces V_j such that $\cdots \subset V_0 \subset V_1 \subset \cdots \subset V_j \subset V_{j+1} \subset \cdots \subset L^2(R^d)$. The function ϕ is known as a *scaling function* or *father wavelet*.

The difference information between two approximations of $f(x)$, $P_j f \in V_j$ and $P_{j+1} f \in V_{j+1}$ is given by the orthogonal projection $Q_j f$ onto the complement W_j of V_j in V_{j+1}. The spaces W_j contain the difference between the information at one resolution and the extra information at a higher resolution in a multiresolution analysis of the function. So we have:

$$V_j \oplus W_j = V_{j+1}$$
$$W_j \perp V_j$$
$$Q_j f = P_{j+1} f - P_j f$$

These spaces W_j are spanned by the wavelets ψ, also known as the *mother wavelet* or *analysing wavelet*. The ϕ and ψ are related. It can be shown that $\{p_n\}$ and $\{q_n\}$ exist such that:

$$\phi(x) = \sum_k p_k \phi(2x-k)$$
$$\psi(x) = \sum_k q_k \phi(2x-k)$$

The ϕ are used for approximations and the ψ for analysing the errors.

Multiresolution analysis can be done in terms of orthogonal or non-orthogonal basis functions. The former are fairly well understood [25]. The most common operations in image analysis however cannot be cast into the orthogonal wavelet framework. Orthogonal wavelets are complex non-symmetric functions whereas "nice" functions generate non-orthogonal wavelets. For this reason we will propose an investigation into non-orthogonal wavelets and multiresolution analysis [16, 17].

Spline Wavelets

A non-orthogonal class of wavelets can be obtained from the cardinal spline functions. In fact typical examples of the scaling functions ϕ introduced above are the cardinal splines illustrated in Figure 3. A cardinal spline is a polynomial spline with equally spaced simple knots [5, 31, 32]. The advantages of splines are:

- The standard basis functions are B-splines that are all convolutions of the unit pulse (the zero order spline in Figure 5).
- Smooth functions with compact support.
- Simple analytic forms that are easy to compute and manipulate.

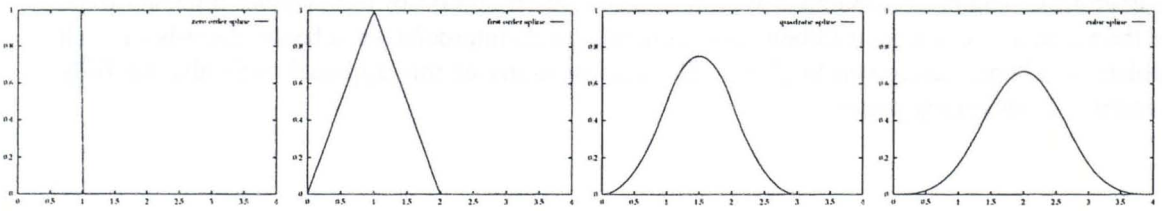

Figure 5: Examples of Cardinal B-Splines.

4.3. Wavelet Compression of Images

Many applications of wavelets to image compression have appeared [2, 14-16, 34]. Some of the most interesting developments have been the papers by Mallat and colleagues who have investigated the "adaptive" compression of images where the compression depends on the detection of important features (edges) and building the compression around these [18, 26, 27]. That this accords with human perceptual predilections can be seen from the ease with which people recognize line drawings of objects which only show the edge features. These methods were able to compress the same image as we used in our example (Figure 4) by factors of between 40 and 100 (≈ 0.081 bits per pixel)..

There is a conjecture by David Marr [28] that if one detects all edges at all the scales then one can reconstruct the image with this information. Wavelets finally allow one to perform this kind of analysis. In [18] on *second generation* image coding the point is made that the same features are called "edges" at one resolution and "textures" at another. Thus if we stop at a particular resolution level the rest of the information that remains (the error term) can be called textures and one can use special encoding methods for those, since the human visual system is not so concerned about the locality of texture elements.

5. Further Development

Utilising the semi-orthogonal *spline wavelets* of Chui [5], we are implementing a wavelet decomposition. The scaling function is the quadratic B-spline; the corresponding spline wavelet is also expressed as a combination of B-splines [5]. Since we have quadratic patches spanning our lattice of data-points (due to the nature of the spline basis) we may take intensity-profile slices parallel to the x-axis and compose a suitable sequence of *X* processor instructions to interpolate these spans. As the resolution gets coarser, our quadratic spans encompass progressively more pixels and we can interpolate scan-lines with comparatively few instructions. Furthermore, due to the perceptual characteristics of the human visual system, we can quantize the *detail signal* coarsely and even set many of these wavelet coefficients to zero, thus making substantial compression gains. The lowest resolution signal can be downsampled (since it is effectively band-limited, having had most of its high-frequency content removed). By the sampling theorem, we can reconstruct the coarse signal and then, utilising our quantized coefficients, synthesize a good approximation to the original.

Initial results, to be reported elsewhere, confirm the robustness of this approach — there is little discernible difference between the original and our reconstructed image. The error present, while negligible, can be eliminated if we adopt a true interpolation of our original intensity data, rather than a more economical (albeit less accurate) quasi-interpolation scheme. Details of such interpolation schemes are given in [7, 8]. The ramifications of this approach will also be fully explored in a forthcoming paper.

6. Conclusion

We have shown how image reconstruction on a simple polynomial basis can be performed at very high speed using the in-house hardware developed for the raster graphics architecture. This should come as no surprise, since also the instructions generated by the area processor (Figure 1) are expressions that describe various splines that are to be interpolated. In [23] we introduced a method for quadratic Phong shading via angular interpolation. The method leads to a parameterized piecewise quadratic expression for $\cos^n\theta$, the cosine of an angle θ ($-\pi/2 < \theta < \pi/2$) raised to a power n ($1 \le n \le 125$). The shape of the curve is very similar to a Gaussian (Figure 6). Using this shading method an area primitive is coded in terms of a spline basis which is to be interpolated by the Difference Engine.

The time the image reconstruction will take does *not* depend on the spacing of the knots in the splines (the lengths of the interpolation spans) but only on the number of knots. An image can be decompressed even at video rates provided that the number of knots are less than the number of pixels to be generated (by some fixed overhead per scanline).

This demonstration now opens the way for using this hardware to reconstruct images that have been coded with a *wavelet transform*. The wavelet transform is a multiresolution description of the image that can be decoded to yield more and more accurate reconstructions of an image. The transform also precisely locates high-frequency features in space and low-frequency signals in the frequency domain. In fact it is argued [14] that wavelet transforms perform better than the discrete cosine transform advocated by the JPEG standard, it fits in better with human perceptual aptitudes and is a more compact coding.

We conclude from this that the Difference Engine may be used to perform the last steps in the process of image synthesis as well as in the process of image reconstruction. This may ultimately lead to display systems in which both the graphics pipeline and an image reconstruction engine

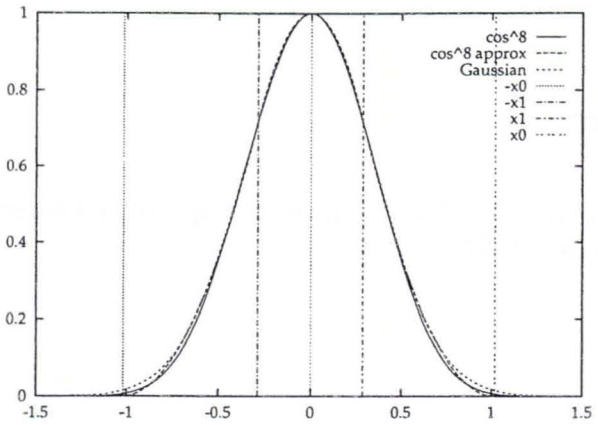

Figure 6: Quadratic Approximation of a Cosine to a Power Used for Phong Shading. *The diagram shows the curve of* $\cos^n$ *and the quadratic spline approximation of the same function, here* $n=8$ *but the approximation is valid for a wide range of* n. *The knot points of the splines are at* $-x_0$, $-x_1$, x_1 *and* x_0. *A Gaussian curve (i.e., a function of the form* e^{-x^2}*) is shown which has the same value at the points* $-x_1$, *0 and* $+x_1$ *as the other curves.*

feed a Difference Engine which performs the necessary spline interpolation (Figure 7).

It is interesting to observe that when Phong shaded images are synthesized with this architecture, that is, when it is used as it was intended, one gets a very efficient encoding of the images in terms of the quadratic [5] spline curve of Figure 3. Moreover the splines are exactly aligned with the edges of the (polygonized) contours in the image. A very efficient single resolution encoding!

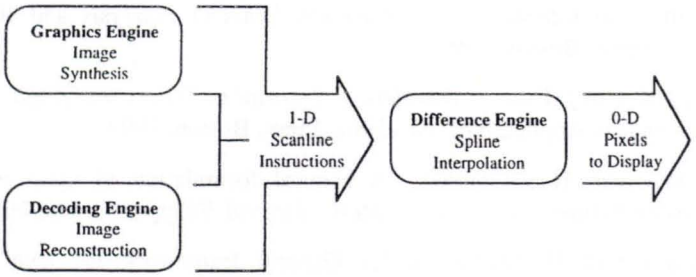

Figure 7: Multi-stream Spline Interpolation System. *The architecture of a display system in which image synthesis and image reconstruction processes share a Difference Engine which performs spline interpolation for the final steps of image generation.*

It is clear that this use of the architecture for decoding wavelet compressed images will be a very useful area for future research. It is apparent that non-orthogonal wavelets seem to be particularly promising. One major area will be adaptive or "intelligent" coding of images that make use of the techniques developed in low level computer vision. One would detect features which are "interesting" for human perception and optimize the coding of these. Only then can the very high compression ratios (> 100) looked for be achieved.

Eventually the system might be used for speech decoding [33, 36], finite element analysis and not just for image decoding. Finally it may be hoped that the investigation into wavelet image

[5] Anti-aliasing produces some complications, notably small pieces approximated by cubic polynomials.

coding techniques may in their turn provide useful spin-offs for image synthesis, one could think of new ways to do adaptive anti-aliasing and of representing textures.

7. Ackowledgements

The authors would like to thank Mr. Patrick Marais of Cape Town University for implementing the wavelet decomposition referred to in Section 5.

References

[1] E.H. BLAKE AND A.A.M. KUIJK, "A Difference Engine for Images with Applications to Wavelet Decompression," in *Proceeding of IMAGE'COM 93*, pp. 309-314, Bordeaux, France, 23-25 March 1993.

[2] A. C. BOVIK, N. GOPAL, T. EMMOTH, AND A. RESTREPO, "Localized Measurement of Emergent Image Frequencies by Gabor Wavelets," *IEEE Trans. Information Theory*, vol. 38, no. 2, pp. 691-712, March 1992. Special Issue on Wavelet Transforms and Multiresolution Signal Analysis

[3] P. J. BURT AND E. H. ADELSON, "The Laplacian Pyramid as a Compact Image Code," *IEEE Trans. Comm.*, vol. 31, pp. 532-540, 1983.

[4] P. J. BURT AND E. H. ADELSON, "A Multiresolution Spline With Application to Image Mosaics," *ACM Trans. on Graphics*, vol. 2, no. 4, pp. 217-236, October 1983.

[5] C.K. CHUI, *An Introduction to Wavelets,* Wavelet Analysis and Its Applications, Academic Press, Boston, 1992.

[6] CHARLES K. CHUI (ED.), in *Wavelets: A Tutorial in Theory and Applications*, Wavelet Analysis and Its Applications, Academic Press, Boston, 1992.

[7] C.K. CHUI AND H. DIAMOND, "A Natural formulation of Quasi-interpolation by Multivariate Splines," *Proc. Amer. Math. Soc*, vol. 99, pp. 643-646, 1987.

[8] C.K. CHUI AND H. DIAMOND, "A General framework for local interpolation," *Numerical Mathematics*, vol. 58(?), pp. 569-580, 1990.

[9] MAC A. CODY, "The Fast Wavelet Transform," *Dr. Dobb's Journal*, vol. 17, no. 4, pp. 16-28, 100-101, April 1992.

[10] I. DAUBECHIES, "Orthonormal Bases of Compactly Supported Wavelets," *Comm. Pure and Appl. Math.*, vol. 41, pp. 909-996, 1988.

[11] I. DAUBECHIES, *Ten Lectures on Wavelets,* CBMS-NSF Series in Applied Mathematics, 61, SIAM, 1992.

[12] I. DAUBECHIES, S. MALLAT, AND A.S. WILLSKY (EDS.), *Special Issue on Wavelet Transforms and Multiresolution Signal Analysis,* IEEE Trans. Information Theory, 38, IEEE, March 1992. Part II of issue number 2

[13] I. DAUBECHIES, S. MALLAT, AND A.S. WILLSKY, "Introduction to the Special Issue on Wavelet Transforms and Multiresolution Signal Analysis," *IEEE Trans. Information Theory*, vol. 38, no. 2, pp. 529-531, March 1992.

[14] P. DESARTE, B. MACQ, AND D.T.M. SLOCK, "Signal-Adapted Multiresolution Transform for Image Coding," *IEEE Trans. Information Theory*, vol. 38, no. 2, pp. 897-904, March 1992. Special Issue on Wavelet Transforms and Multiresolution Signal Analysis

[15] R.A. DEVORE, B. JAWERTH, AND B.J. LUCIER, "Image Compression Through Wavelet Transform Coding," *IEEE Trans. Information Theory*, vol. 38, no. 2, pp. 719-746, March 1992. Special Issue on Wavelet Transforms and Multiresolution Signal Analysis

[16] J.-C. FEAUVEAU, "Nonorthogonal Multiresolution Analysis Using Wavelets," in *Wavelets: A Tutorial in Theory and Applications*, ed. C.K. Chui, Wavelet Analysis and Its Applications, pp. 153-178, Academic Press, Boston, 1992.

[17] H.G. FEICHTINGER AND K. GROCHENIG, "Non-Orthogonal Wavelet and Gabor Expansions, and Group Representations," in *Wavelets and Their Applications*, ed. M.B. Ruskai, pp. 353-375, Jones and Bartlett, Boston, 1992.

[18] J. FROMENT AND S. MALLAT, "Second Generation Compact Image Coding with Wavelets," in *Wavelets: A Tutorial in Theory and Applications*, ed. C.K. Chui, Wavelet Analysis and Its Applications, pp. 655-678, Academic Press, Boston, 1992.

[19] M.A. GURAVAGE, E.H. BLAKE, AND A.A.M. KUIJK, "XInPosse: Structural simulation for graphics hardware," in *Advances in Computer Graphics Hardware, VI*, ed. A. Kaufman, Springer-Verlag, Berlin, 1992.

[20] P.J.W. TEN HAGEN, A.A.M. KUIJK, AND C.G. TRIENEKENS, "Display Architecture for VLSI-Based Graphics Workstations," in *Advances in Computer Graphics Hardware, I*, ed. W. Strasser, Springer-Verlag, Berlin, 1987.

[21] J.A.K.S. JAYASINGHE, A.A.M. KUIJK, AND L. SPAANENBURG, "A display controller for an object-level frame store system," in *Advances in Computer Graphics Hardware, III*, ed. A.A.M. Kuijk, pp. 141-170, Springer-Verlag, Berlin, 1990.

[22] J.A.K.S. JAYASINGHE, G. KARAGIANNIS, F. MOELAERT EL-HADIDY, O.E. HERRMANN, AND J. SMIT, "Two-level pipelined systolic array graphics engine," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 3, pp. 229-236, 1991.

[23] A.A.M. KUIJK AND E.H. BLAKE, "Faster Phong shading via angular interpolation," *Computer Graphics Forum*, vol. 8, no. 4, 1989. Please note that on p. 321 the definitions of a and b should be swapped.

[24] A.A.M. KUIJK, E.H. BLAKE, AND P.J.W. TEN HAGEN, "An Architecture for Interactive Raster Graphics," *Proceedings of the Seventh Eurographics Workshop on Graphics Hardware*, Cambridge, U.K., 1992.

[25] S. MALLAT, "A Theory of Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674-693, 1989.

[26] S. MALLAT AND W.L. HWANG, "Singularity Detection and Processing with Wavelets," *IEEE Trans. Information Theory*, vol. 38, no. 2, pp. 617-643, March 1992. Special Issue on Wavelet Transforms and Multiresolution Signal Analysis

[27] S. MALLAT AND S. ZHONG, "Wavelet Transform Maxima and Multiscale Edges," in *Wavelets and Their Applications*, ed. M.B. Ruskai, pp. 67-104, Jones and Bartlett, Boston, 1992.

[28] DAVID MARR, in *Vision*, W.H. Freeman, 1982.

[29] MARY BETH RUSKAI (ED.), in *Wavelets and Their Applications*, Jones and Bartlett, Boston, 1992.

[30] G. STRANG, "Wavelets and Dilation Equations: A Brief Introduction," *SIAM Review*, vol. 31, pp. 614-627, 1989.

[31] M. UNSER AND A. ALDROUBI, "Polynomial Splines and Wavelets — A Signal Processing Perspective," in *Wavelets: A Tutorial in Theory and Applications*, ed. C.K. Chui, Wavelet Analysis and Its Applications, pp. 91-122, Academic Press, Boston, 1992.

[32] M. UNSER, A. ALDROUBI, AND M. EDEN, "Fast *B*-Spline Transforms for Continuous Image Representation and Interpolation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 3, pp. 277-285, 1991.

[33] M.V. WICKERHAUSER, "Acoustic Signal Compression with Wavelet Packets," in *Wavelets: A Tutorial in Theory and Applications*, ed. C.K. Chui, Wavelet Analysis and Its Applications, pp. 679-700, Academic Press, Boston, 1992.

[34] R. WILSON, A.D. CALWAY, AND E.R.S. PEARSON, "A Generalized Wavelet Transform for Fourier Analysis: The Multiresolution Fourier Transform an Its Application to Image and Audio Signal Analysis," *IEEE Trans. Information Theory*, vol. 38, no. 2, pp. 674-690, March 1992. Special Issue on Wavelet Transforms and Multiresolution Signal Analysis

[35] A.P. WITKIN, "Scale-space Filtering," in *Int. Joint Conf.on Artificial Intelligence*, pp. 1019-1022, Karlesruhe, Germany, 1983.

[36] X. YANG, K. WANG, AND S.A. SHAMMA, "Auditory Representation of Acoustic Signals," *IEEE Trans. Information Theory*, vol. 38, no. 2, pp. 824-839, March 1992. Special Issue on Wavelet Transforms and Multiresolution Signal Analysis