Pergamon

Pattern Recognition, Vol. 31, No. 8, pp. 1179–1190, 1998

0031-3203/98 $19.00 + 0.00

PII: S0031-3203(97)00130-1

AN IMPROVED PIECEWISE APPROXIMATION ALGORITHM FOR IMAGE COMPRESSION†

JOSEPH MODAYIL,‡ HOWARD CHENG‡ and XIAOBO LI*

Department of Computing Science, University of Alberta, Canada T6G 2H1

(*Received* 22 *May* 1997; *in revised form* 9 *October* 1997)

Abstract—One-dimensional lossy compression schemes, such as piecewise approximation with triggers (PAT) (Walach and Karnin, *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Vol. 1, pp. 529–532, Tokyo, Japan, April 1986) have the advantage of being computationally simple. Several previous papers have examined this algorithm and a variety of changes have been proposed. This paper adds new features and also incorporates some ideas from previous papers to arrive at a modified version: MPAT. The modifications vary parameters for the trigger function and the thresholds. Furthermore, they also introduce new features such as context modelling, different interpolations, and early triggers. These changes add flexibility, decrease the compressed image size, and improve the reconstruction quality, while maintaining a complexity advantage over other algorithms. A complexity count is also performed to quantitatively demonstrate the benefits of this algorithm over alternatives. © 1998 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

One dimensional Lossy image compression Fractal PAT MPAT
Complexity Piecewise linear approximation

1. INTRODUCTION

Recently, digital images have been expanding in usage and size. This has added pressure to improve image compression algorithms so that they produce smaller file sizes with a lower computational requirement. Algorithms based on DCT or wavelet transforms[1] can reduce file size, but there is still a need for computationally simple methods. This paper examines modifications to a one-dimensional compression scheme, piecewise approximation with triggers (PAT).[2] This is altered to create modified piecewise approximation with triggers (MPAT) which retains PAT's advantage of being computationally simpler than conventional two-dimensional methods. Although neither algorithm will compress images as well as two-dimensional methods, both can be implemented on systems with little processing power, such as video cameras, mobile wireless communication systems, and handheld computers.

A one-dimensional image compression algorithm has two parts—the image is scanned into a one-dimensional array and then the resulting array is compressed. The scan employed is the Hilbert[3] scan as suggested previously,[4] which is known to maintain interpixel correlation well.[5] Previous work[6] has shown that the Hilbert scan is beneficial and that it can be successfully generalized to work on images of arbitrary dimensions. After the scan is performed, the PAT algorithm compresses the resulting array by an approximation technique. The compressed output is entropy coded using an arithmetic coder[7] with multiple models—a different model for each type of data. The use of arithmetic coders does not significantly increase the complexity.

The original idea has been examined and altered by other papers; however, a comprehensive examination of the entire algorithm has been lacking. This paper attempts to fill the void by amalgamating and improving previous ideas, and introducing new changes where appropriate. The result is called MPAT, and it builds on the basic PAT algorithm by adding options and modifications to improve image quality and operational flexibility. To justify further research in this area, a thorough analysis and complexity count are performed.

The paper is organized as follows. In Section 2, the PAT algorithm is introduced. In Section 3 the modifications to create MPAT are described. In Section 4, the performance of both algorithms is considered, and they are compared to previous results. In Section 5, the complexity of the algorithms is discussed. In Section 6, the results are summarized.

2. PIECEWISE APPROXIMATION WITH TRIGGERS (PAT)

PAT is a lossy image compression scheme that operates in the spatial domain. It offers good performance at very low complexity levels. The concept was originally given by Walach and Karnin.[2] It evolved

* Author to whom correspondence should be addressed.
† This research is supported in part by the Motorola Wireless Data Group and the Canadian Natural Sciences and Engineering Research Council under Grant OGP9198.
‡ Supported by the SCP '96 scholarship.

from the idea of "laying a yardstick" down on a one-dimensional signal and recording the horizontal distance the yardstick traveled. Since a yardstick has a fixed length, knowing the horizontal distance travelled implies a mimium vertical distance has been covered. This is how the signal is compressed; instead of transmitting the height of the signal at each point, the signal is assumed to be lying on the yardstick. Although the horizontal distance dictates a vertical displacement, it does not indicate if the signal went up or down; the sign of the height change must also be transmitted. The decoder can perform a lossy reconstruction using only the horizontal distances and the signs of the heights. The decoder creates a piecewise linear approximation to the original signal.

Using the model of a yardstick, horizontal distance and height would be related by the formula $x^2 + y^2 = r^2$. This was generalized to use a trigger function, TF, which takes a horizontal distance and returns a height. Then the transmitted distance becomes the smallest distance at which the change in signal height exceeds the trigger function, namely

$$\begin{cases} \min\limits_{1 \leqslant i \leqslant i_{\max}} \{i : |s_{x+i} - r_x| > TF(i)\} & \text{and} \\ \qquad\qquad \text{if such an } i \text{ exists,} \\ i_{\max} \qquad\qquad \text{otherwise,} \end{cases}$$

where $i_{\max}$ is the maximum length, s_x is the signal value at x, and the reconstruction at x is r_x. The sign of $s_{x+i} - r_x$ is transmitted after each distance is sent (called a trigger).

A diagram showing the behaviour of the algorithm is in Fig. 1. The original discrete signal is indicated by small black dots connected by a broken line. The encoder and decoder are assumed to know the starting point precisely. The encoder sends the distance at which the signal first falls outside the envelope (dotted line) provided by the trigger function. The encoder then transmits a sign indicating whether the signal went through the top envelope or the bottom. The decoder uses the horizontal distance travelled, along with the sign, to find the location where the signal exceeded the trigger function, and creates a reconstructed point there. The decoder can linearly interpolate between the orignal point and the reconstructed point. At this time, both the encoder and decoder agree on a common point and the process iterates.

A simple modification to the above procedure is suggested by Karnin and Walach, which involves the use of thresholds. When the distance to be transmitted is one and $|s_{x+1} - r_x| > Threshold$, then it is likely that the signal is near an edge. A zero distance is transmitted followed by a three-bit (eight level) quantized value. The value of *Threshold* is stored in $TF(0)$. In effect, $TF(0)$ acts as a threshold to determine if a different region of the image is being entered. Using different models for distances, signs, and quantized values, the output is compressed with an arithmetic coder. The original paper suggests the use of vertical line subsampling to increase performance, so that only every second or third vertical line is included during the scan. This introduces significant visual distortion; hence it is not used in our experiments. For the decompression phase, the signal is linearly interpolated between the reconstructed values. The encoding algorithm for an array A of 8-bit integers is stated below to facilitate complexity analysis. The encoder and decoder maintain synchronization via *start*. This is especially important after a threshold, as the decoder will normalize the three-bit value. Note that *start* can never fall out of the range of the image, so bound checking is not required when determining *start* values.

PAT encoding algorithm

Assume that the image has been Hilbert scanned into the array A.
Output the first element of the array A in $A[i_{start}]$
Set $start = A[i_{start}]$.
Let the last element to be processed be in i_{end}.
Set $i = 1$.
repeat
  $diff = |start - A[i_{start} + i]|$
  while ($diff \leqslant TF(i)$ and $i < i_{\max}$ and $i_{start} + i < i_{end}$)
    $i = i + 1$
    $diff = |start - A[i_{start} + i]|$
  if ($i = 1$ and $diff > TF(0)$)
    Output the distance 0.
    Output the three most significant bits of $A[i_{start} + i]$.
    $start = (A[i_{start}]$ AND 11100000) OR 00010000
  else
    Output the distance i.
    Output $sign$ = the sign of $A[i_{start} + i] - start$.
    if ($sign$ is positive)
      $start = start + TF(i)$
    else
      $start = start - TF(i)$
  $i_{start} = i_{start} + i$
  $i = 1$
until ($i_{start} = i_{end}$)

3. MODIFIED PAT (MPAT)

Some changes can be made to the original PAT algorithm to improve its results and to add flexibility. Using a fixed trigger function yields a file size that cannot be well predicted. However, file size is important because PAT is not designed for progressive transmission; a truncated file does not produce a good approximation to the original. One area for improvement is to vary parameters to allow the file size and the distortion to be controlled. Another area for improvement is to introduce new features which can improve the image quality by altering aspects of the algorithm. We propose the following changes to PAT. All graphs in this section show results from tests performed on the Lena image (512×512).

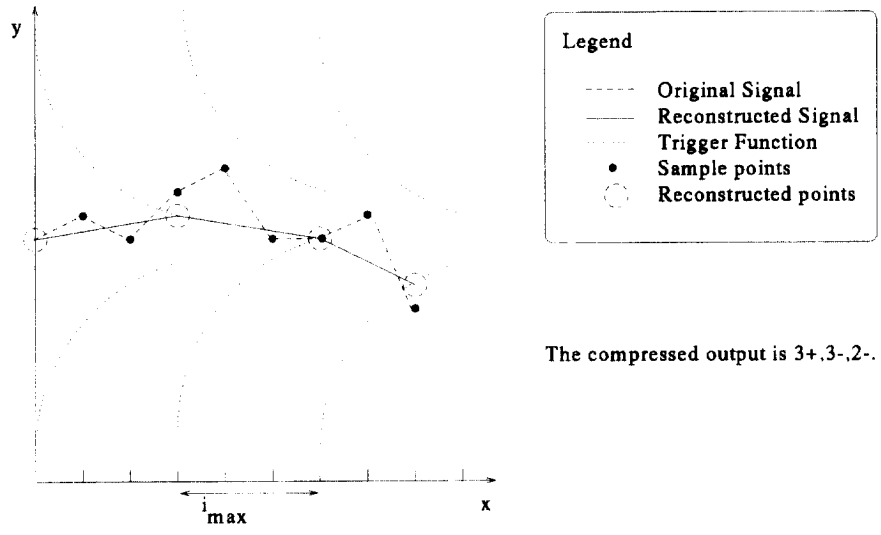

Fig. 1. PAT encoding of a sample signal.

3.1. *Varying parameters*

3.1.1. Trigger function. Earlier papers[8,4] on PAT generally used the trigger function originally suggested,[2] namely

$$TF(x) = 15151311975444444444.$$

In our experiments a wide variety of exponential trigger functions,[2,5] were examined from the set of functions

$$TF(x) = ae^{bx} + c,$$

where a, b, and c are parameters. These functions performed well without the use of subsampling as originally proposed, and their exponential structure may be well adapted to the logarithmic response of the human visual system. These functions also offered a systematic way to generate trigger functions which permitted the performance of the algorithm to be tested at a variety of bit rates, a feature which previously had not been available.

From the set of functions classified by $TF(x) = ae^{bx} + c$, the members from the subset $TF(x) = ae^{-0.05x} + 2$ were found to be near optimal on rate-distortion graphs. The dependence on these particular values is not critical, but a negative b value with small magnitude performed much better than those with large magnitudes. The results obtained for this paper were all generated by this subset of trigger functions. To create a fair comparison between the PAT and MPAT algorithms, the same trigger functions were used in each bank of tests.

Another change was to set i_{max} to 64, as opposed to 16 which was used in other papers. The advantage of a large i_{max} value is that the algorithm can handle large smooth regions with fewer triggers, and it permits compression at low bit rates. Although a large i_{max} value requires more bits per trigger, the use of an arithmetic coder reduces this penalty. The results from varying the i_{max} values is given in Fig. 2. Setting i_{max} to 32 seems to yield almost identical results to PAT for Lena, but on other images this is not true. Benefits of using a smaller i_{max} value include fewer models and symbols for the arithmetic coder, and smaller tables for interpolation (see Interpolation Methods).

3.1.2. Threshold. Thresholds are required to avoid degradation of sharp edges in a signal. A few iterations may be required to transmit a large signal change when using a small trigger function. This smooths out edges and wastes many bits on a poor approximation. The use of thresholds permits rapid changes without high overhead.

Previous papers did not examine changes to the thresholds. Two aspects of thresholds must be considered. The first is deciding when it should occur. The *Threshold* value was varied linearly with $TF(0)$ and the results are shown in Fig. 3. The results show that the *Threshold* value provided by the trigger function, namely $TF(0)$, performs adequately.

The second aspect is the number of bits to allocate for a thresholded value. A minimum for this is implied by *Threshold*. Since $|r_x - s_{x+1}| > TF(0)$, at least $256/(2 \times TF(0))$ states are required to ensure that the use of the threshold is beneficial. Then for a particular choice of $TF(0)$, the minimum number of bits required to hold the thresholded value must be at least

$$\left\lceil \log_2\left(\frac{256}{2 \times TF(0)}\right) \right\rceil.$$

Allocating less than the above number of bits makes a threshold less accurate than required to guarantee that it is beneficial. Using more bits is a waste, since the human eye is less sensitive to variations at edges.

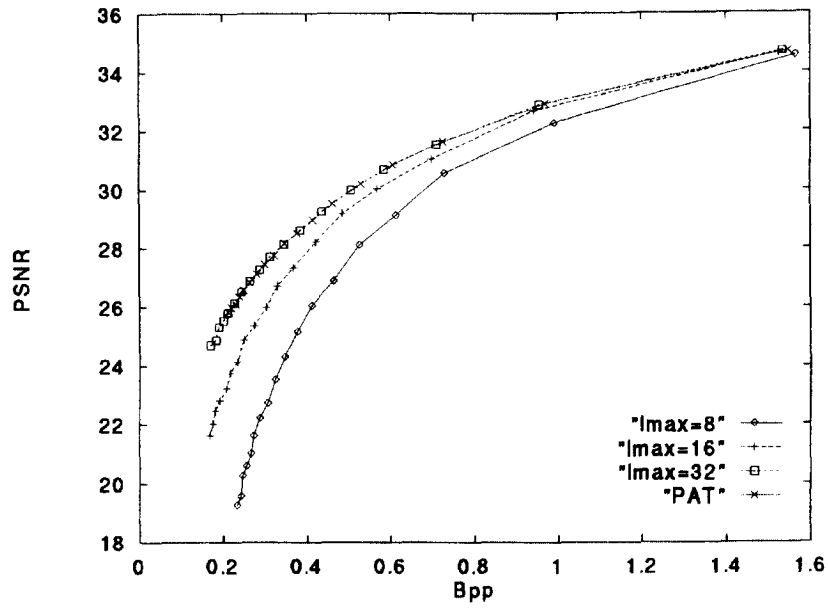

Fig. 2. Varying i_{max} on Lena ($i_{max} = 64$ in PAT).

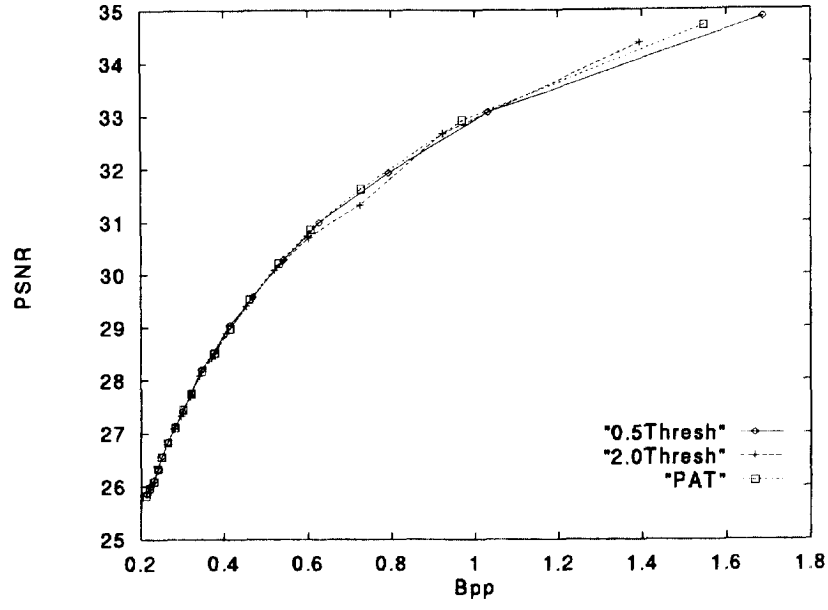

Fig. 3. Varying the threshold. PAT corresponds to $Threshold = TF(0)$, and $xThresh$ corresponds to $Threshold = x \times TF(0)$.

Hsu and Pai[8] suggest using all eight bits, but this is extremely costly when using small *Threshold* values which cause many thresholds to be transmitted. The original choice of three bits caused thresholds to degrade the image at high bit rates, so PAT was implemented with the modified bit allocation for thresholds.

3.2. *New features*

3.2.1. Context modeling. Context modeling was introduced so that the model used to encode the current distance is based on the last transmitted distance. Previous papers had not examined this potential area of compression. The additional information permits the arithmetic coder to better model the incoming data. This choice of model can be made by a table lookup, which incurs negligible computational overhead; complicated methods of selecting a model were avoided due to computational cost. To choose a model, the last distance transmitted is placed on a logarithmic map. Namely, for a fixed parameter K, two distances d_1 and d_2 will select the same model if and only if there exists $n \in \mathbf{N}$ such that $K^n < d_1 < K^{n+1}$

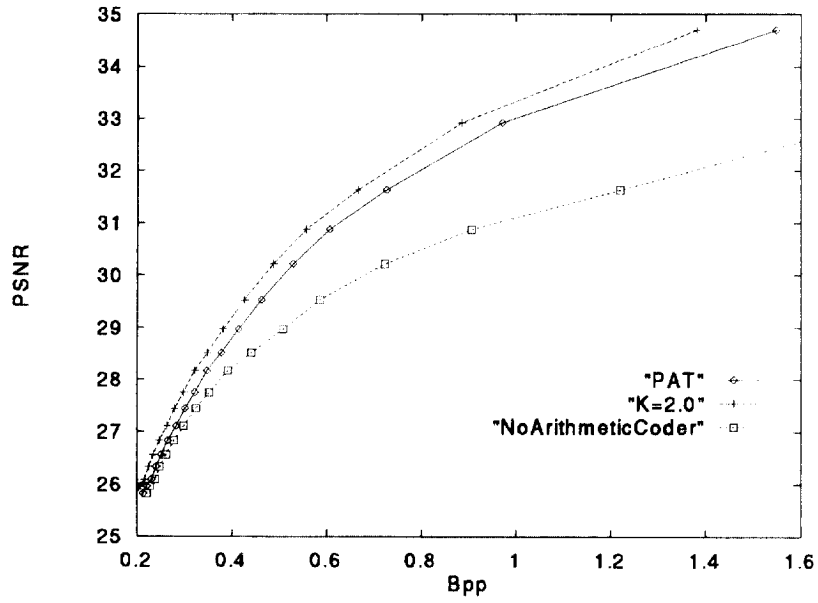

Fig. 4. Effects of entropy coding on Lena. Results with PAT (no context modeling), full context modeling, and no entropy coding are shown.

and $K^n < d_2 < K^{n+1}$. The choices for K were varied from 1 to 8, but the performance was only marginally altered. The best performance was observed in the region about $K = 2$.

Context modeling can be further increased by coding the current sign with a model based on the previous sign. The thresholds can also be coded using different models based on the most significant bits of the most recent *start* value. The number of models for thresholds was chosen to be the number of possible threshold values—the function for choosing a threshold value to be transmitted is used on the last *start* value to choose the model. This yields good performance at high bit rates. Without the use of arithmetic coders this advantage is entirely lost. The results of using the added models, and the results without arithmetic coding are displayed in Fig. 4. Clearly, arithmetic coding greatly enhances performance, but the poor performance without arithmetic coding is exaggerated by the large $i_{\max}$ value chosen for PAT.

3.2.2. Interpolation method. A variety of interpolation methods were examined. Previous papers had used either linear or flat interpolation. The most obvious method is to use Linear interpolation, and that is the method used in PAT. Other methods include Flat, Quadratic, Flat + Linear, and Flat + Quadratic. The formulas below give the interpolations for n values between $A[i_{start} + 1]$ and $A[i_{start} + n]$ inclusive. The performance of the various modifications is shown in Fig. 5.

Flat interpolation:

$$A[i + i_{start}] = A[i_{start}], \quad 1 \le i < n,$$
$$A[n + i_{start}] = A[i_{start}] + sign \times TF(n).$$

A Flat interpolation will copy the last reconstructed value over the distance of interpolation and place the new reconstructed value in the last position. This has the advantage of being very fast as only one addition is required per interpolation. However, blocking artifacts appear at low bit rates with this method.

Linear interpolation:

$$A[i + i_{start}] = A[i_{start}] + (i)\,(sign \times TF(n)/n) \quad 1 \le i < n.$$

With Linear interpolation, the blocking effects are not as significant. Indeed, only at very low bit rates are blocks visible with this method. The values that are added to $A[i_{start}]$, namely $(i)\,(sign \times TF(n)/n)$, can be generated at run time, or can be pre-computed. Having smaller $i_{\max}$ values will reduce the memory overhead of having pre-computed tables. The Linear interpolation method is simple but requires an addition for each interpolated pixel which is the most significant cost. This is the method used in the PAT algorithm.

Quadratic interpolation:

$$A[i + i_{start}] = A[i_{start}] + (i^2)\,(sign \times TF(n)/n^2), \quad 1 \le i < n.$$

In the tests, this method performed marginally better than any of the other methods. Like Linear interpolation the values that must be added to $A[i_{start}]$ can be computed at run time or can be pre-computed. If it is pre-computed it is no more expensive at run time than Linear interpolation—its superior performance comes at no extra cost.

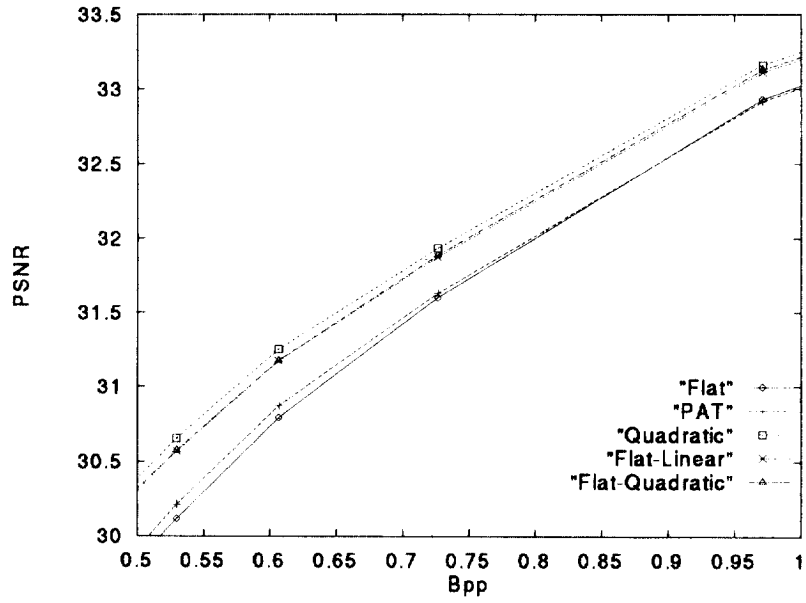

Fig. 5. Varying the interpolation method for Lena.

Flat + Linear:

$$A[i + i_{start}] = A[i_{start}], \quad 1 \le i \le \lfloor \tfrac{n}{2} \rfloor,$$

$$A[i + i_{start} + \lfloor \tfrac{n}{2} \rfloor] = A[i_{start}] + (i)(sign \times TF(n)/m),$$

$$m = n - \lfloor \tfrac{n}{2} \rfloor, \ 1 \le i \le m.$$

Since Flat reduces the run-time computation and Linear performs well, a successful attempt was made to merge the two methods. The first-half of the interpolation was performed as Flat, and the latter half was linearly interpolated. This interpolation outperforms both Linear and Flat methods on the rate-distortion graph, while maintaining computational simplicity. This is evident in Fig. 5. An added bonus is that pre-computed interpolation tables can be halved, reducing memory overhead.

Flat + Quadratic:

$$A[i + i_{start}] = A[i_{start}], \quad 1 \le i \le \lfloor \tfrac{n}{2} \rfloor,$$

$$A[i + i_{start} + \lfloor \tfrac{n}{2} \rfloor] = A[i_{start}] + (i^2)(sign \times TF(n)/m^2),$$

$$m = n - \lfloor \tfrac{n}{2} \rfloor, \ 1 \le i \le m.$$

Given the success of the Flat + Linear approach it is logical to consider combining Flat and Quadratic in a similar manner. Unfortunately, it does not result in a dramatic rise in performance. In fact, it does not perform better than Quadratic, which suggests that a level of diminishing results has been reached. Still, if values are to be pre-computed, it offers slightly better performance over the Flat + Linear approach at no extra cost.

3.2.3. Early Trigger. Another modification to improve edge handling is to have an early trigger.[8] If $i > 1$ and $diff > Threshold_{Early}$ when a trigger is ready to be sent, then the distance i is decremented by one. The value for $Threshold_{Early}$ can be determined experimentally as a proportion of $TF(0)$. It was found that setting $Threshold_{Early} = 2 \times TF(0)$ performed well at bit rates around 1, as is shown in Fig. 6. This modification allows the reconstruction to stop before large differences are encountered, permitting sharper edges. It also permits the reconstruction to adapt better to the signal. In the PAT algorithm, it takes at least one pixel before the algorithm can recognize an edge, and this was visibly noticeable in the PAT method.

The use of early triggers also permits guarantees to be made about the approximation. The original signal is guaranteed to be within the corresponding trigger function value up to the trigger distance, and within $Threshold_{Early}$ at the trigger distance. A threshold will occur only if the new *start* value is further than *Threshold* from the signal. This does well in detecting actual edges while ignoring noise. Since the reconstruction on a threshold is guaranteed to be within *Threshold* of the original signal, any point on the reconstruction can be no further than $Threshold_{Early}$ from the original.

A similar method was proposed in reference (8). However, it always uses an early trigger, and does not perform as well with the chosen trigger functions. Although an early trigger requires a higher bit rate for a given trigger function, the rate-distortion graph shows the corresponding increase in quality is worth the cost. The advantage of using an early trigger selectively compared to using an early trigger always is shown in Fig. 7.

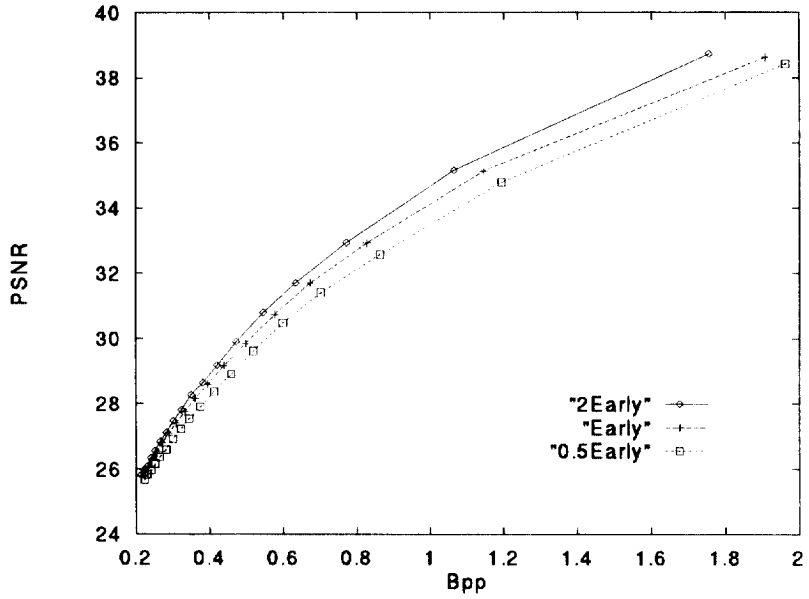

Fig. 6. Varying $Threshold_{Early}$ on Lena. xEarly corresponds to $Threshold_{Early} = x \times TF(0)$.

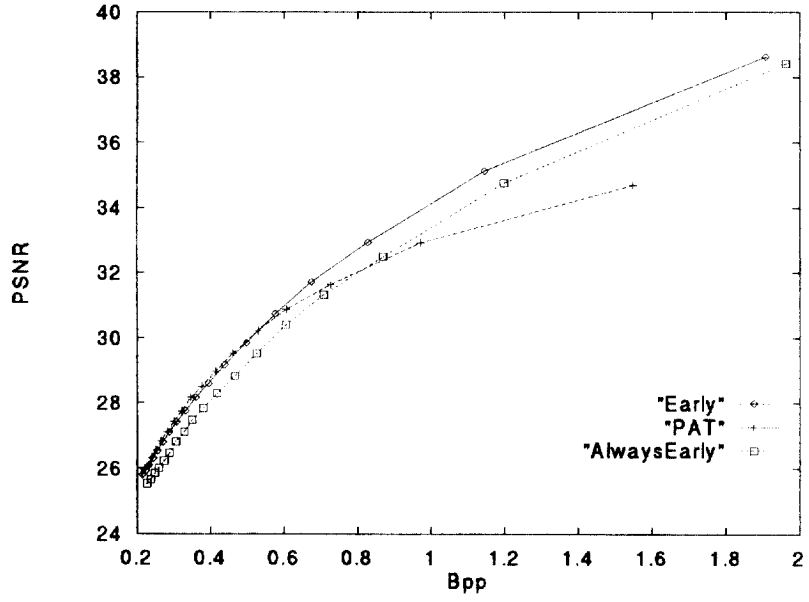

Fig. 7. Comparing usage of early triggers on Lena. Early corresponds to $Threshold_{Early} = TF(0)$, PAT represents using no early trigger, and AlwaysEarly reresents using early triggers everywhere.

Although this modification is almost transparent to the decoder, it does introduce a slight complication. With the addition of this change, the reconstructed values can fall outside the range of the original image. To avert this, the decoder must perform bound checking on each trigger. The coder needs only to check for this event when an early trigger is performed. If it falls out of range, the reconstruction must be altered so that it is bounded. This complicates the interpolation as a pre-computed table cannot be used for these triggers. Fortunately, this event does not occur frequently in most images.

4. PERFORMANCE EVALUATION

Here we examine the performance of MPAT, which is formed from PAT and modifications discussed in the previous section. MPAT consists of PAT combined with $TF(x) = ae^{-0.05x} + 2$, $i_{max} = 64$, $Threshold_{Early} = 2 \times TF(0)$, $Threshold = TF(0)$, Flat + Linear interpolation, and context modeling with $K = 2$. The version of PAT used in these tests consists of the original PAT combined with the use of arithmetic coding, modified bit allocation for thresholds, the choice of $i_{max} = 64$, and the use of Hilbert scanning.

Table 1. Results of PAT on Lena (512 × 512)

Parameter a	10	20	30
Bpp	0.971	0.607	0.462
PSNR	32.92	30.87	29.53
Thresholds	17,126	6059	2463
Triggers	31,040	21,818	16,993

Table 2. Results of MPAT on Lena (512 × 512)

Parameter a	10	20	30
Bpp	0.962	0.579	0.433
PSNR	35.51	32.10	30.40
Thresholds	20,595	6957	2754
Normal triggers	26,584	20,355	16,412
Early triggers	5097	1649	669
Flat pixels	113,039	122,460	125,597

These adjustments to PAT were included to better demonstrate the effects of the modifications in MPAT.

The counts of various events in the algorithms in Tables 1 and 2 are included to demonstrate typical results. In these tables thresholds are not counted as triggers, and a normal trigger is one that is not early. Clearly, the number of triggers and thresholds that have to be coded by the arithmetic coders is small; hence, arithmetic coding is not a significant penalty. The same trigger functions produce files of comparable sizes in the two algorithms, but the MPAT algorithm has much higher PSNR values. The ratio of thresholds to triggers is seen to rise with increasing bit rates. This would suggest that for trigger functions with small a, a special symbol for the distance 0 would improve results if arithmetic coding is not employed. The total number of early triggers remains relatively small compared to the number of normal triggers in MPAT, but they are likely responsible for the rise in the number of thresholds between PAT and MPAT. The number of triggers in PAT and MPAT are very close—this shows that early triggers do not generate excessive triggers. From the table, the average distance traveled by a trigger in MPAT at 0.579 bpp is $512^2/(20355 + 1649) \approx 11.9$ which justifies a large i_{max} value. The number of Flat pixels in Table 2 represents the number of pixels which were handled by the Flat portion of the Flat + Linear interpolation. It is close to half of the number of pixels in the image—the difference arises from the floor operation in Flat + Linear interpolation and thresholds. For all three trigger functions, the reconstructions never fell out of the (0–255) range of the image.

The results given compare favourably with those reported previously. The results of Sorek and Zeevi[5] did not match our expectation as their SNR and bpp ratio (30.5 SNR at 0.375 bpp) placed higher than MPAT. When implemented by the authors, their algorithm yielded a PSNR of 28.09 at 0.455 bpp, and so the discrepancy is likely due to a different choice of measure.

The tables do not accurately represent the visual quality of MPAT. With the MPAT algorithm on Lena at approximately 0.5 bpp, the reconstruction errors introduce noise. This is detectable but does not detract visually from the image, as can be seen in Figs 8 and 9. In fact, this noise is often not noticeable in photographic images. The reconstructed images tend to be "sharp" instead of "blurry" as can be observed in the reconstructions of some transform-based techniques.[9] If a small two-dimensional smoothing filter is applied to the reconstruction, the results are subjectively better as the sharpness in homogeneous regions is decreased, but filters can be computationally expensive and so were not used. When trigger functions with large a values are used, the noise becomes visually distracting and blocking effects occur. This is because large regions are approximated by a single interpolation, which causes details to be missed. The choice of the trigger function to minimize the bit rate without causing visual degradation is only determinable visually. Although the

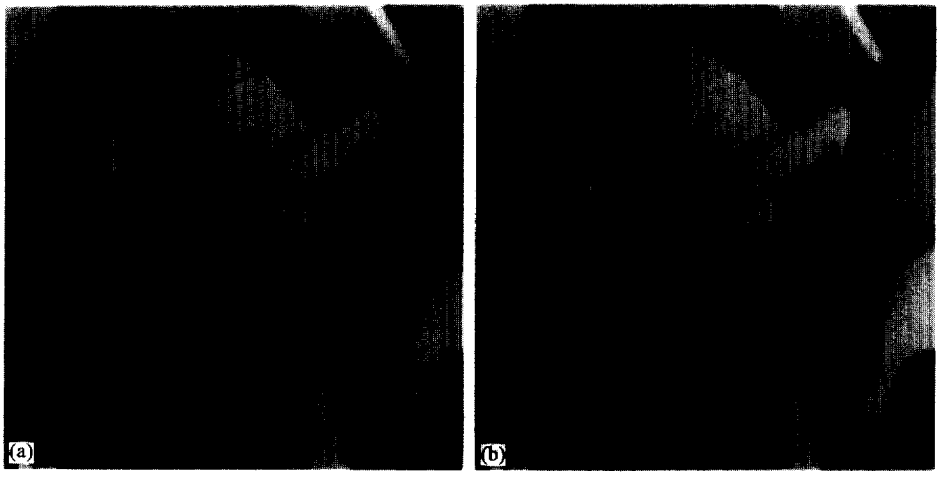

Fig. 8. Lena (512 × 512). (a) original; (b) MPAT reconstruction at 0.499 bpp (PSNR = 31.24).

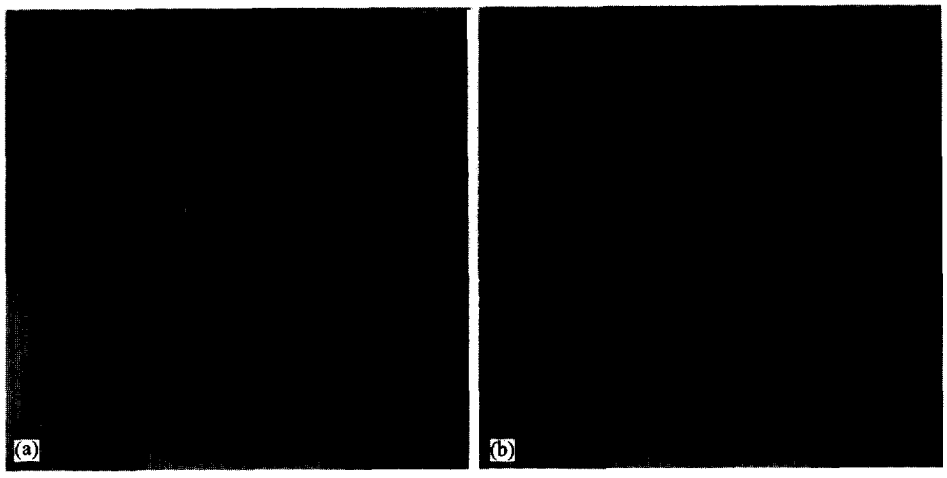

Fig. 9. Baboon (512 × 512). (a) original; (b) MPAT reconstruction at 0.680 bpp (PSNR = 23.10).

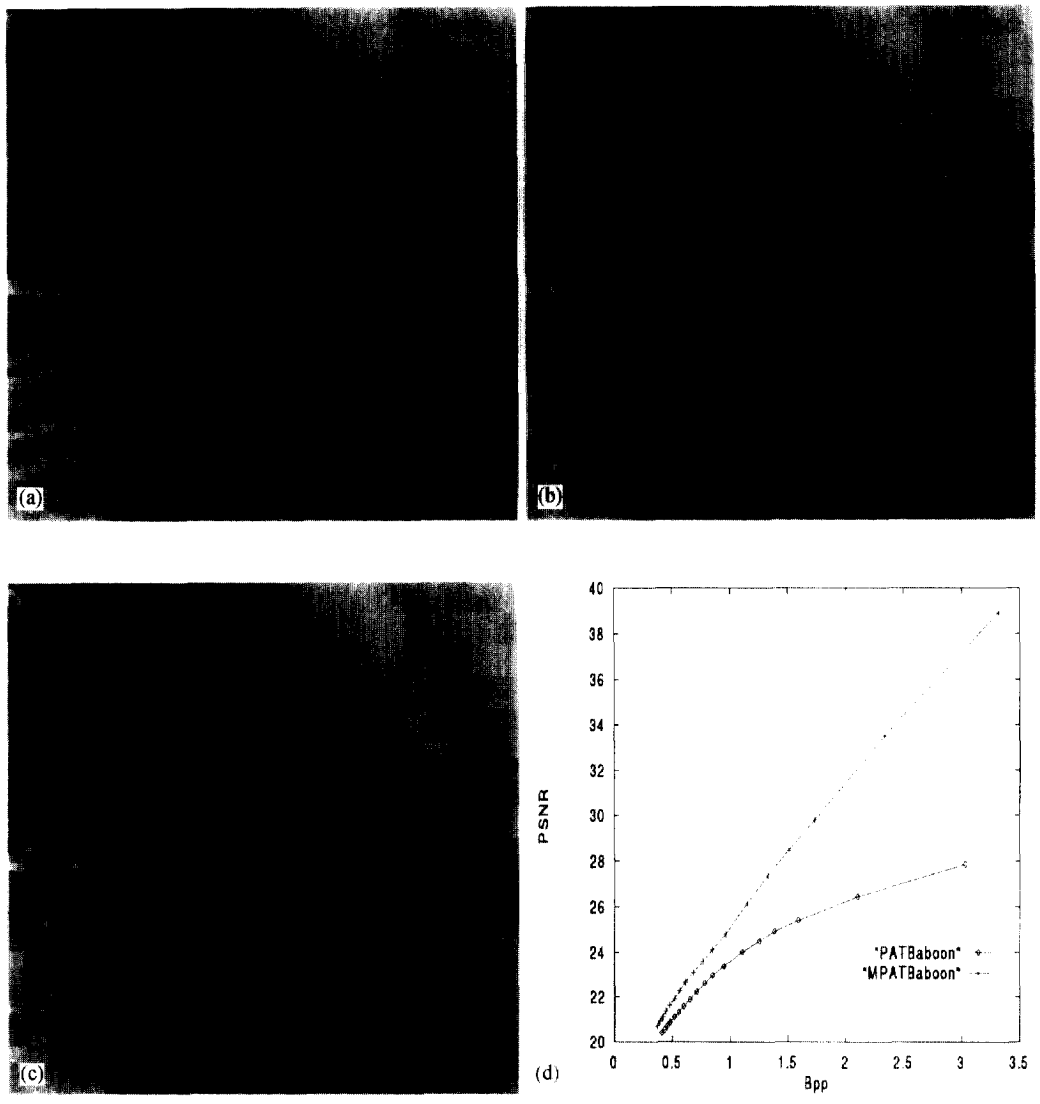

Fig. 10. Zoom on left of mouth of Baboon (512 × 512). (a) original; (b) PAT reconstruction at 0.709 bpp (PSNR = 22.23); (c) MPAT reconstruction at 0.680 bpp (PSNR = 23.10); (d) rate-distortion graph.

PSNR values for a set of images can be extremely low, the images may still be visually acceptable, as shown in Fig. 9.

One cause for the discrepancy between visual and numerical results is that the reconstruction can simulate details such as hair without a good match to the original image. As can be seen in Fig. 10, when regions with hair are being compressed, the reconstructed signal will oscillate, but it may not do so at the same location or to the same extent as the original. The visual results of the PAT are poorer than that of MPAT because it does not react as quickly to edges—much of the baboon's hair is lost. The rate-distortion graph for the baboon image shows that the modifications are extremely effective at higher bit rates.

The fact that both PAT and MPAT can operate on the baboon image speaks of the robustness of the algorithms. This is due in part to the removal of noise. In the original version of the baboon, some noise can be seen intermingled with the hair. In the reconstruction for PAT and MPAT, the noise is removed or diminished. The cost is the removal of fine details, which is seen in Fig. 11 where the eyelashes do not exist in either reconstruction.

With MPAT, the guarantee of performance allows the same trigger functions on two vastly different images to yield similar PSNR values at high bit rates. The PAT version makes no such guarantees as the graphs shown in Figs 10d and 11d. PAT makes clearly defined edges jagged, whereas MPAT preserves them, as shown in Fig. 11.

Previous work[6] has shown that while PAT does not compress images as well as wavelet methods, the file sizes are comparable at low bit rates. With the added modifications, MPAT performs approximately

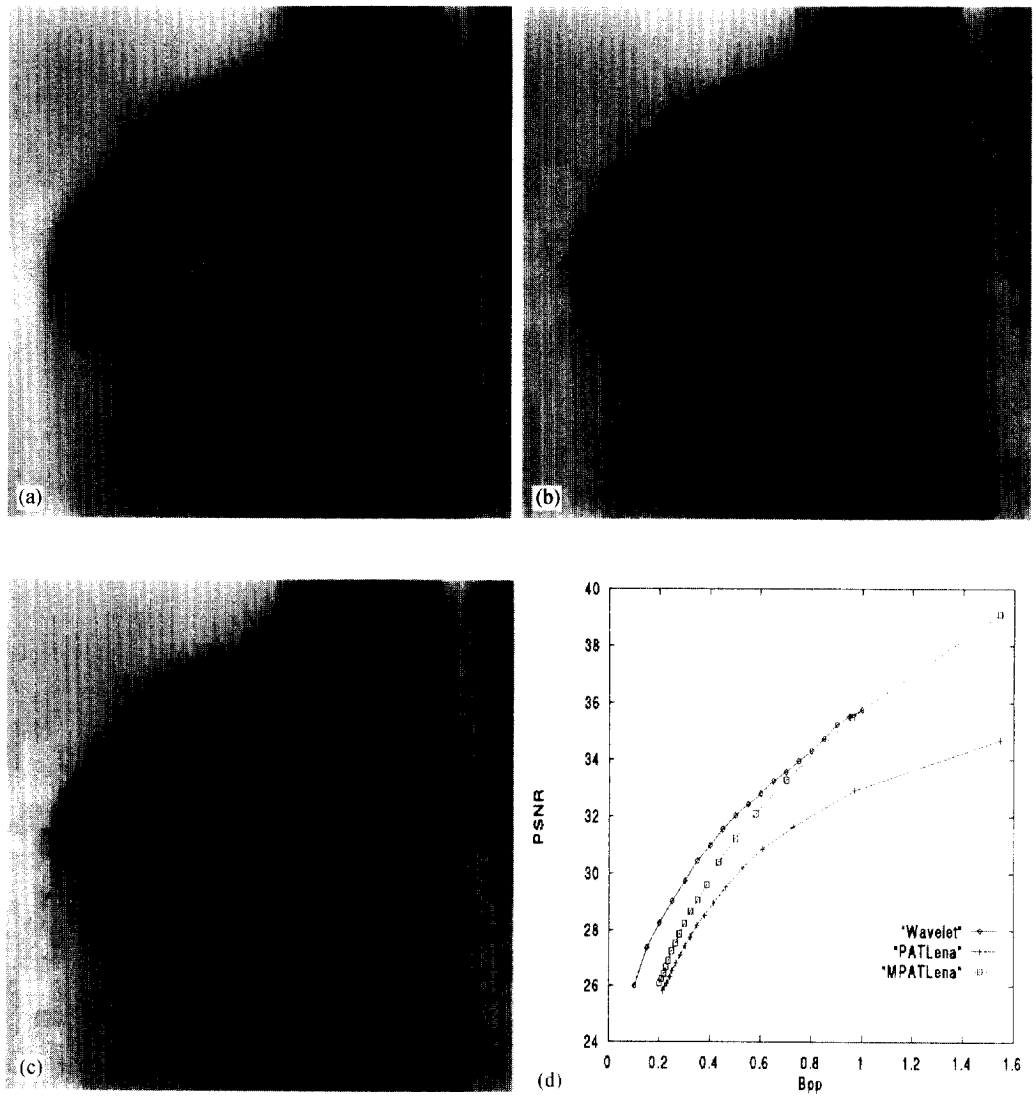

Fig. 11. Zoom on right eye of Lena (512 × 512). (a) original; (b) PAT reconstruction at 0.530 bpp (PSNR = 30.21); (c) MPAT reconstruction at 0.499 bpp (PSNR = 31.24); (d) rate-distortion graph.

Table 3. PAT operation counts

Category	Additions	Comparisons	Shifts
Compression			
Per pixel	1	2	0
Per threshold	0	3	1
Per trigger	2	3	0
Decompression			
Per threshold	0	2	1
Per trigger of distance $N > 0$	$N + 2$	$N + 3$	0

Table 4. MPAT operation counts

Category	Additions	Comparisons	Shifts
Compression			
Per pixel	1	2	0
Per threshold	0	3	1
Per regular trigger	2	4	0
Per early trigger	2	5	0
Decompression			
Per threshold	0	2	1
Per trigger of distance $N > 0$	$N - \lfloor \frac{N}{2} \rfloor + 1$	$N + 4$	1

as well as a wavelet based algorithm on Lena at 1 bit per pixel, which is shown in Fig. 11d. The wavelet algorithm is a multi-resolution one-dimensional algorithm.[6] For the wavelet results, the picture was read into an array using the Hilbert scan, and then compressed using multi-resolution analysis on the array. While one-dimensional wavelet compression does not perform as well as its two-dimensional counterparts, it provides a benchmark for one-dimensional compression. Essentially, MPAT matched the performance of multiresolution analysis: an impressive feat given the simplicity of the MPAT. This shows that MPAT is a strong compression method, even in comparison to some better known algorithms.

5. COMPLEXITY ANALYSIS

The complexity of PAT-based algorithms is extremely low, as stated by previous papers, and demonstrated here. The operation counts were performed on versions of PAT and MPAT which had been significantly optimized for speed. For these counts, increment operations and simple array references were ignored. Operations required to perform the scan, and those used by the arithmetic coders were also ignored. For simplicity, subtractions were counted as additions, and divisions as multiplications. All comparisons in the algorithms were counted, including those in loop structures. Table lookups were assumed for all interpolations (though the boundaries comparisons were still performed) as the reconstruction rarely falls out of the range of the image.

In Tables 3 and 4, the operations used during compression and decompression are given. Since table lookups are used for interpolation, only integer operations are required throughout the algorithm. The only modifications which are reflected in the complexity analysis are early triggers and interpolation method. Context modeling is handled by table lookup, while thresholds and trigger functions are predetermined.

Table 5 shows the operations required by the algorithms at approximately 0.8 bpp. The results for both PAT and MPAT were obtained using the trigger function

$$TF(x) = 14e^{-0.05x} + 2.$$

Also included in the comparisons are the counts for a computationally simple two-dimensional quadtree algorithm at 0.8 bpp.[10] From Table 5, it is evident that PAT-based algorithms are significantly simpler than quadtree in terms of computation. It is worth noting that the quadtree counts were taken from an implementation that was aiming for computational simplicity. The number of comparisons in the quadtree algorithm is not given,[10] but it is logical to assume that it is significantly more complex than PAT-based algorithms. The low comparison counts are included to demonstrate that the control logic is also very simple. These tables show the benefits of PAT-based algorithms, as quadtree algorithms are already among the most computationally efficient of two-dimensional methods.[1] From Table 5, we can say that MPAT and PAT are an order of magnitude simpler than two-dimensional analogues. In decomposition, it can also be seen that MPAT is simpler than PAT in that it has far fewer additions to perform.

Table 5. Operations per pixel for Lena (512 × 512)

Category	Additions	Multiplications	Comparisons	Shifts
Compression				
PAT (0.80 bpp)	1.203	0	2.428	0.041
MPAT (0.78 bpp)	1.203	0	2.570	0.048
Quadtree (0.80 bpp)	2.13	0.16	Not given	0.50
Decompression				
PAT (0.80 bpp)	1.161	0	1.345	0.041
MPAT (0.78 bpp)	0.603	0	1.461	0.152
Quadtree (0.80 bpp)	6.82	3.00	Not given	0.16

6. CONCLUSIONS

The MPAT algorithm consistently outperforms PAT in image quality. The family of trigger functions considered in this paper performed well in comparison to other reported results. Also introduced was the set of trigger functions $TF(x) = ae^{-0.05x} + 2$ which performs near optimally within its class. This set of functions reduces a previous inadequacy of the PAT algorithm which did not give a systematic method for attaining variable compression ratios. The threshold paramaters were varied to improve performance. Context modeling was introduced to improve compression ratios and performance. Early triggering was introduced which allows edges to be more accurately reconstructed. The interpolation model was changed to Flat + Linear to yield a result which outperforms Linear interpolation in quality and complexity. A great benefit of PAT-based algorithms is an extremely low computational requirement. The only real drawback is the lack of a precise control on file size. This may not be too important, however, as the PSNR yielded by a trigger function can be fairly accurately predicted. Despite its low complexity, MPAT managed to match a one-dimensional wavelet algorithm at high bit rates. MPAT has promise for applications in portable units with little processing power and which require moderate compression.

REFERENCES

1. R. Clarke, *Digital Compression of Still Images and Video*. Academic Press, New York (1995).
2. E. Walach and E. Karnin, A fractal based approach to image compression, in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Vol. 1, pp. 529–532, Tokyo, Japan, April (1986).
3. D. Hilbert, Ueber stetige abbildung einer linie auf ein flächenstück, *Math. Ann.* **38**, 459–460 (1891).
4. K. Yang, L. Wu and M. Mills, Fractal based image coding using peano scan, in *ISCAS'88*, Vol. 3, pp. 2301–2304, Espoo, Finland, June (1988).
5. N. Sorek and Y. Zeevi, On-line visual data compression along a one-dimensional scan, in *Visual Communications and Image Processing'88*, Vol. 1001, pp. 764–768, SPIE, (1988).
6. J. Modayil, H. Cheng and X. Li, An examination of one-dimensional image compression techniques, in *Proc. 4th IEEE Int. Conf. Multimedia Computing and Systems*, (1997).
7. I. H. Witten, R. M. Neal and J. G. Cleary, Arithmetic coding for data compression, *Comm. ACM* **30(6)**, 520–540 (1987).
8. C. Hsu and C. Pai, Fractal based image coding using the sharp edge treatment, in *Proc. IEEE Int. Symp. on Circuits and Systems*, Vol. 1, pp. 291–294, Chicago, Illinois, May (1993).
9. B. Moghaddam, K. Hintz and C. Stewart, Space-filling curves for image compression, *SPIE Automat. Object Recognition* **1471**, 414–421 (1991).
10. J. Knipe and X. Li, A new quadtree decomposition reconstruction method, *Proc. 13th Int. Conf. on Pattern Recognition*, pp. B364–369, August (1996).

About the author JOSEPH MODAYIL is currently a graduate student in mathematics at the University of Alberta, Canada. His research interests include image processing, and differential geometry.

About the author—HOWARD CHENG is currently a graduate student in computing science at the University of Alberta, Canada. His research interests include image processing and algebraic computation.

About the author—XIAOBO LI received the Ph.D. degree in Computer Science from the Michigan State University. His research interests include pattern analysis and image processing. He is currently a professor at the University of Alberta, Edmonton. He served as program co-chair of the Vision Interface '92 conference, and he is an associate editor for *Pattern Recognition*. He is a Senior Member of IEEE.