

A pattern-based lossy compression scheme for document images

QIN ZHANG AND JOHN M. DANSKIN

*Dartmouth College
Department of Computer Science
6211 Sudikoff Laboratory
Hanover, NH 03755, USA*

e-mail: {zhangq, jmd}@cs.dartmouth.edu

SUMMARY

CDIS is a pattern based, nearly lossless compression system for scanned document images. In this paper, we introduce a hierarchical lossy pattern instance position coding technique which results in a significant improvement in compression with no visible artifacts. CDIS codes text positions by automatically formatting blocks of text, then transmitting the position errors for each pattern. Lossy coding is achieved by coding errors in reduced precision, subject to quality guarantees.

KEY WORDS Document processing Document image Compression

1 INTRODUCTION

For our purposes, document images, or textual images, are black and white (binary) images containing mostly aligned text. Compression is desirable for document image transmission and storage. Without compression, a letter size document image when sampled at 300 dots per inch would contain about 2 Megabytes of data, even though the image may contain only a few lines of text. If we sent this image over telephone lines at 28,800 bits/sec, it would take almost 5 minutes.

Traditional methods of document image compression exploit the statistical dependence between neighboring pixels. For instance, the run-length coding used by the CCITT (International Telephone and Telegraph Consultative Committee) standard [1], and the context-based predictive coding used in the JBIG (Joint Bilevel Image Experts Group) standard [2]. Since these methods exploit only pixel level dependences of the image, they fail to provide good compression for document images.

Document image compression based on pattern matching is an effective technique for text pages. Document images contain patterns such as characters and lines, which may be repeated many times in the image. Pattern matching coding techniques exploit these macroscopic properties of document images. These techniques were originally proposed in [3] and further studied in [4–6]. [6] and [7] discuss lossless compression based on pattern matching.

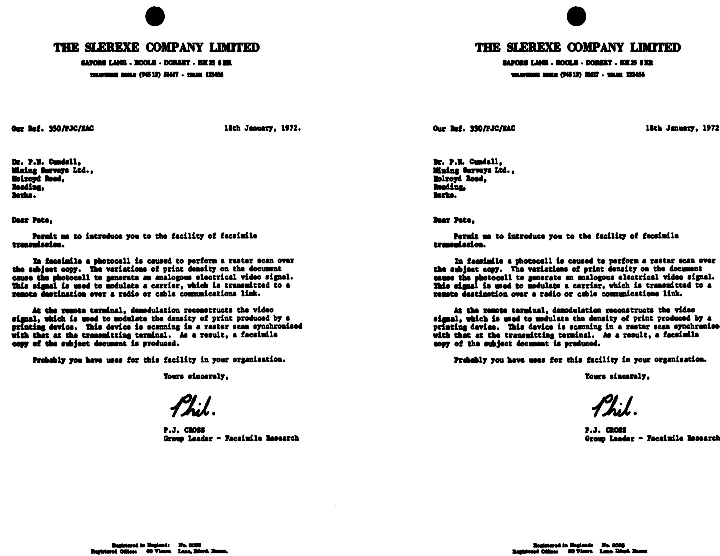


Figure 1. The original (left) and the reconstructed image of CCITT1 at 95:1 compression.

Our Compression for Document Image System (CDIS) is also pattern-matching based. Exploiting the knowledge of document images one step further, CDIS takes advantage of the structural layout of document images. This system effectively codes pattern sequences and positions in the image by coding the pattern positions hierarchically: First it divides the text image into blocks and automatically formats the text within each block by estimating each pattern's position. Then it transmits the position error (the difference between the pattern's actual and estimated positions) for each pattern in reduced precision. In nearly-lossless mode, CDIS compresses the same test document set (single page) about 13% more than the best previous results. Better results (18% saving) are obtained on multi-page documents.

CDIS reproduces document images in nearly-lossless mode. The reconstructed image is an approximation to the original image. The quality of the reconstructed images is guaranteed by conservative pattern matching and position coding methods. CDIS's pattern matching method ensures that only closely matched patterns are substituted from the original images. Its pattern position coding method ensures that the position of each pattern is coded within a few pixels of its original position. Furthermore, CDIS provides the option to change and control the degree of accuracy used to code pattern positions. Figure 1 shows an example original document and its lossy reconstruction. Figure 2 compares a zoomed portion of the two images.

2 PREVIOUS WORK

Image compression based on pattern matching works in this way: patterns, which are characters from an image, are extracted from the image to be compressed. They are compared and matched to previously transmitted patterns. If a match is detected, only the position of the pattern and the index of the pattern are transmitted. This process greatly reduces

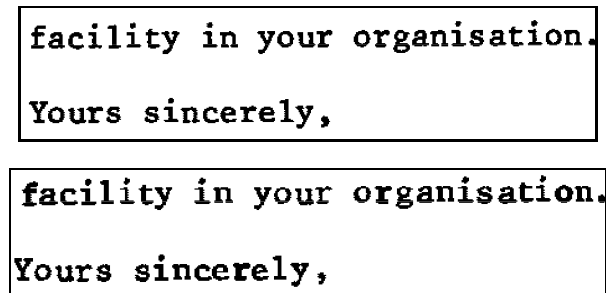


Figure 2. A zoomed portion of the original (top) and the reconstructed image of CCITT1 (bottom): some patterns in the reconstructed image are substituted, and the x positions of the patterns are not exactly the same as in the original image. The loss, however, is strictly controlled by CDIS's error correction method.

the number of bits required to code the patterns. During the compression process, a library that contains one representative of each pattern class is gradually built up, and the patterns in the library are compared with the patterns to be encoded. This compression scheme is lossy: a reconstructed document image is slightly different from the original image due to the pattern matching and substitution process. This algorithm can be made lossless by coding the difference between the original image and the reconstructed image.

Ascher and Nagy [3] first proposed document image compression based on pattern matching. Their system uses fixed-length codes to represent pattern sequences and positions, achieving about 16:1 compression.

Pratt *et al.*'s combined symbol matching (CSM) [8] combines exact coding and pattern-matching within the same document. Symbols are removed from the document image and coded by pattern-matching. The remainder of the image is termed the *residue* and encoded exactly using a traditional binary image compression scheme. CSM compresses symbol positions using a variable-length static coding scheme. [9] and [4] also used variable-length codes to represent the pattern positions.

Mohiuddin [7] studied pattern matching algorithms and developed a lossy/lossless compression system. His lossless compression scheme uses the matching pattern in the library as part of the context to code each new pattern bitmap. This scheme compresses images much better than conventional context-based predictive coding schemes, such as JBIG.

Witten *et al.* described a two-stage lossy/lossless compression system, Textual Image Compression (TIC), for document images [6]. TIC uses the Prediction by Partial Matching (PPM) modeling and arithmetic coding [10] to code pattern sequences. TIC codes the offset between one pattern and the next, instead of the absolute positions. These offsets are coded using a first-order predictive model based on the pattern index. TIC uses a lossy version of the reconstructed image to build contexts for predictive coding of the lossless document image. In lossy mode, TIC has the best compression ratio of all previous document image compression schemes; in lossless mode, TIC has results similar to Mohiuddin's lossless image compression system.

Witten *et al.* also experimented with inserting a special pattern to represent all kinds of white space between patterns. These white space patterns were intended to improve

compression of pattern sequences. Witten *et al.* concluded, however, that this method helped their performance only by little and was not worth the effort.

3 BETTER COMPRESSION FOR DOCUMENT IMAGES

Our Compression for Document Image System (CDIS) goes one step further to exploit higher level properties of document images and obtain improved compression.

A lower bound for the size of a compressed document image is the entropy of the text in the image and the geometric and typographic information. Lossy TIC compresses document images into the smallest size of all the image compression systems. But the compressed images contain roughly 40 bits/character while the entropy of English text is around 2 bits/character and the cost of formatting information is a few bits per character for regular documents at most, leaving room for further work in this area.

A typical pattern-based image compression system encodes an image in three components:

- The pattern library: a set of small bitmap patterns.
- A pattern index sequence representing the text in the image.
- The position of each pattern in the image.

For a single-page document, more than 50% of the compressed file is bitmap images. The proportion of bits due to the bitmap images is small for multi-page documents, because many patterns appear on multiple pages. A typical application of document image compression systems deals with multi-page documents. In a multi-page document, pattern indices and positions make up the majority of the code space, so it is important to code them compactly. CDIS emphasizes better compression of pattern positions.

3.1 Rebuilding text source

CDIS rebuilds text source: it processes the pattern index sequence so that the sequence is as close to the original text source as possible. To rebuild the text source, first we find and extract the marks (connected blobs of ink), then we arrange the marks in the same order as the original text source. Some previous document image compression systems, such as TIC, also perform these steps. The resulting pattern sequence, however, is not close to the original text source for two reasons: first, when a logical pattern is built from physically disjoint components, such as i , it is extracted as two separate marks; second, spaces and line breaks, important parts of the original text source, have no representation in the resulting sequence. CDIS adds two extra tasks to the text rebuilding process. First, it merges some patterns; second, it inserts two special patterns, *space* and *new block*, to represent the spaces between words, and line breaks respectively.

CDIS uses Johnsen *et al.*'s mark boundary tracing method [9] to isolate and extract the patterns from an image, as does TIC. In TIC, if a pattern has two disconnected parts, as i does, it is extracted as two separate patterns. Usually, when a logical pattern is built out of physically disjoint components, the components are horizontally aligned and vertically displaced as in i , $.$, and j . CDIS detects this configuration and merges it into a single pattern, reducing the number of library patterns and making the pattern sequence as close to the source as possible.

CDIS also introduces two special patterns: the *space* pattern and the *new block* pattern. The *space* pattern is inserted between adjacent words to represent the space between words.

The *new block* pattern is placed at the end of each text line and between horizontally adjacent patterns which are separated by an unusually large gap. This special pattern appears at the end of a line, at the end of a paragraph, and between two columns in a line.

CDIS uses two thresholds, H_1 and H_2 , to decide where to insert *space* and *new block*. CDIS measures the spaces between adjacent patterns. If the space is larger than H_2 , a *new block* is inserted between the patterns; if the space is between H_1 and H_2 , a *space* is inserted. CDIS calculates H_1 and H_2 by seeking bimodality in this distribution of space widths between adjacent patterns. Two peaks in the distribution are the inter-character space and the inter-word space. H_1 separates the inter-character space from the inter-word space, and H_2 separates the inter-word space from the inter-block space. Thresholds are recalculated for each new block. This local threshold calculation method is necessary because multiple fonts may be used in the same document. A set of thresholds that accurately distinguish inter-character spaces from inter-word spaces, and inter-word spaces from inter-block spaces in one font may not be accurate in another.

Figure 3 shows where CDIS inserts *space* and *new block* patterns in a typical document: CCITT1. CDIS correctly identifies most inter-word spaces. CDIS also successfully identifies blocks. As you can see, each line of text is usually marked as one block. In two lines where there is an unusually large space between patterns, two blocks are defined.

The addition of the *space* and *new block* patterns, and the merging of parts of the same pattern make the pattern sequence closer to the text source; PPM compression works better on this rebuilt pattern sequence than it does on the same pattern sequence without pattern merging and spaces, especially on multi-page documents. Furthermore, these methods make the lossy compression of pattern positions possible, which greatly reduces the required code space.

3.2 Lossy compression of pattern positions


CDIS achieves better compression of pattern positions by coding them hierarchically and inexactly. The method works in three steps:

1. CDIS divides a document image into *blocks* and encodes the exact position of each *block*.
2. CDIS automatically formats the text in each block by guessing the relative horizontal position of each pattern within a block.
3. CDIS transmits the errors between the predicted positions and the actual positions at some predefined resolution.

CDIS uses an adaptive arithmetic coder [10] to code x position errors, and y positions relative to each block.

A block is a pattern sequence terminated by the *new block* pattern. Within a block, patterns are placed in a horizontal line with a small amount of space, w_s , between them. There are also inter-word spaces, w_s , which are indicated by the *space* pattern index. In CDIS's current implementation, a *new block* is always inserted at the end of a line. In a regular one column article, each line of text is a block. The title and each line of an address are also blocks. Figure refc12 shows the blocks in CCITT1.

Once the blocks in the image are identified, the positions of the blocks are encoded and transmitted. CDIS does not encode the absolute positions of the blocks; it guesses the



THE SLEREXE COMPANY LIMITED
 (SAPORS LANE, BOOLE, DORSET, BH 25 3 ER)
 (TELEPHONE BOOLE (94513) 51617, TELEX 123456)

(Our Ref. 350/PJC/BAC) (16th January, 1972.)

(Dr. P.N. Cundall,
 (Mining Surveys Ltd.,
 (Holroyd Road,
 (Reading,
 (Berks.)

(Dear Pete,)


(Permit me to introduce you to the facility of facsimile
 (transmission.)

(In facsimile a photocell is caused to perform a raster scan over
 (the subject copy. The variations of print density on the document
 (cause the photocell to generate an analogous electrical video signal.
 (This signal is used to modulate a carrier, which is transmitted to a
 (remote destination over a radio or cable communications link.)

(At the remote terminal, demodulation reconstructs the video
 (signal, which is used to modulate the density of print produced by a
 (printing device. This device is scanning in a raster scan synchronised
 (with that at the transmitting terminal. As a result, a facsimile
 (copy of the subject document is produced.)

(Probably you have uses for this facility in your organisation.)

(Yours sincerely,)



(P.J. CROSS)
 (Group Leader - Facsimile Research)

(Registered in England. No. 3008)
 (Registered Office) (80 York, Lane, Wood, Essex)

Figure 3. Block boundary and Space patterns in CCITT1: [and] define block boundaries. An underline represents a space pattern. Usually a block is one line of text, but if there is an unusual space between two patterns, such as in the line above the address, an extra block is defined. CDIS inserts space correctly, except the signature line, where the pattern positions are irregular. Nonetheless, the pattern positions are recovered due to error correction.

position of the next block from the current block and codes the errors using a zero order predictive model and an arithmetic coder.

CDIS guesses the x -positions of patterns in a block. First, the number of spaces, and the total width are measured and transmitted. Then, the CDIS image decoder calculates the average inter-character space w_s from the width of the block w_d , the total width of patterns w_p , and the average width of inter-word spaces w_S which are known to the decoder, because $w_d = w_p + w_S * S + w_s * (p - 1)$, where S is the number of *space* patterns and p is the number of patterns in the block. w_s is the default x offset between adjacent patterns. *Space* patterns are treated as other patterns in the block but with a fixed width of w_S .

CDIS measures the error between the estimated x position and the actual x position. This error is then transmitted at an adjustable resolution. CDIS chooses a reasonable resolution to ensure the quality of the reproduced image. When CDIS codes the error exactly, the pattern position is also reconstructed exactly. For most of our experiments, we coded position errors at half resolution, so that patterns in the reconstructed image are shifted by at most one pixel. Reconstructed x positions of the patterns are approximations to the original. For document images, the spaces between letters are usually even, especially within a block. It is hard to visually distinguish between the original images and the reconstructed images with the lossy pattern position coding. See [Figures 1, 2, 5, 6 and 7](#) to compare original and reconstructed images.

CDIS encodes the y offsets of patterns from a block baseline. Because each pattern usually occurs in the same position relative to the block baseline, CDIS codes y offsets using a first order predictive model indexed by the pattern index. Y offsets are usually coded with greater resolution than x offsets, because our experiments with position error show that the eye is much more sensitive to errors in this [dimension](#). [Figure 4](#) shows what happens when we allow single pixel errors in the y direction.

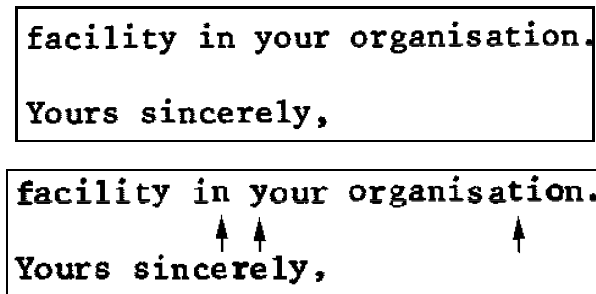


Figure 4. A zoomed portion of the original (top) and the reconstructed image (bottom) with single pixel errors in the y position: the errors are especially apparent for the marked patterns.

The algorithmic complexity of our position coding scheme is $O(n)$ (where n is the number of patterns in the image). The constant factor is high because CDIS makes three passes over the pattern sequence. In the first pass, CDIS determines where to insert *space* and *newblock* patterns and merges the two parts of a logical pattern. In the second pass, CDIS codes the position of each block and autoformats the text inside each block. In the third pass, CDIS codes position errors. It is possible to reduce the constant factor by making fewer passes. We plan to do this in the future.

4 EXPERIMENT

We built two versions of CDIS: “lossless” CDIS, which transmits position errors exactly (The system is still lossy since matched and replaced patterns are not always the same), and lossy CDIS, which transmits the x errors at half of the original resolution so that any pattern may shift its position in the reconstructed image by one pixel horizontally. We tested lossy CDIS’s performance against lossy TIC, which is part of the public-domain Managing Gigabytes (MG) system [11] and the best previous lossy document image compression system. In this test, CDIS and TIC share the same pattern matching algorithm, CSIS.

We ran CDIS and TIC on the same test document set and measured the compression ratio for each system. We also compared the size of the pattern position portion of the compressed image files, to determine which system compresses pattern positions more compactly. Finally, we counted the number of characters in two test documents to compute the average number of bits per character after compression. This per character sum is further broken down into three components: pattern library bits, pattern index bits, and pattern position bits. These bit counts help us to determine which components of the compressed image take up code space and where possible improvements to CDIS might be.

We chose ten single-page documents from CCITT’s standard test images and the MIT AI lab’s *classic hit* technical report collection for our test. We also tested a multi-page document: pages 2-4 of the *BROOKS* document also from MIT’s *classic hits*. The resolution of all of the included images is 300 dpi. We used these images because they are available on the Internet and well-scanned. CCITT images and *classic hits* can be obtained from the Finnish University and Research network (<ftp.funet.fi>) and publications.ai.mit.edu/classic-hits via ftp. The images are a mix of business letters and technical papers, so the compression ratio on these images is a reasonable indication of the performance of each system. Figures 1, 5 and 6 show original and reconstructed images from our test set.

5 RESULTS

Table 1 shows CDIS’s performance on single-page documents. As you can see, CDIS’s average output is only about 87% of TIC’s average output. CDIS achieves more than 4 times the compression of group 4 FAX coding.

Figure 1 shows the CCITT1 original, and a copy reconstructed after 95:1 compression by CDIS. Figure 2 shows details from these two images, allowing a closer comparison. Figures 5, 6, and 7 also allow comparisons between original and reconstructed images. It is quite difficult to see the difference between original and reconstructed images due to nearly-lossless quality control of the encoder.

Table 2 compares the compression performance of lossy CDIS, lossless CDIS, and TIC. As you can see, CDIS achieves three to six times TIC’s compression performance on pattern positions. Pattern positions are generally about 20% of TIC’s output on single-page documents. In lossless mode, CDIS’s code size is about half of TIC’s while providing the same quality of reconstructed images.

Table 3 shows CDIS’s performance on multi-page documents. *BROOKS* is a three-page document. Compression ratios are higher than for the single-page documents for both systems because the pattern library can be reused for multiple pages. CDIS’s performance advantage is more pronounced in this case since the proportion of pattern positions in the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
 ARTIFICIAL INTELLIGENCE LABORATORY

A. I. Memo 809 May, 1988

ACHIEVING ARTIFICIAL INTELLIGENCE
 THROUGH BUILDING ROBOTS

Rodney A. Brooks

Abstract. We argue that generally accepted methodologies of Artificial Intelligence research are limited in the proportion of human level intelligence they can be expected to realize. We argue that the currently accepted decomposition and static representation used in such research are wrong. We argue for a shift to a process based model, with a decomposition based on task solving behavior as the organizational principle. In particular we discuss building robotic teams.

Acknowledgments. This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the research is provided in part by an IBM Faculty Development Award, in part by a grant from the Systems Development Foundation, and in part by the Advanced Research Projects Agency under Office of Naval Research contract N00014-86-K-0021.

© Massachusetts Institute of Technology 1986

AI through Building Robots

1. Introduction

There has been considerable philosophical debate on the possibility of "human level" artificial intelligence, centered around the notion that it requires as background the totality of previous which make up the human way of being in the world (Dreyfus 71, 81).

In this note we use a technical rather than philosophical argument that machines must indeed have a rich background of experience of being if they are to achieve human level intelligence. Unlike Dreyfus however, we conclude that (artificial) intelligent behavior is achievable with computers without the aid of heuristics, common-sense, or other holistic mechanisms. Rather, by adopting an incremental construction approach, progress towards this goal can be expected soon. (Naturally the author and his students are currently following this enlightened path.)

2. Three observations about the failure of AI.

In this section we give three reasons why AI has failed to live up to its early promise, and moreover why it will not live up to its current promise.

A. A lesson from evolution

We already have an extensive proof of the possibility of intelligent earlier-human beings. Additionally many animals are intelligent to some degree. (This is a subject of intense debate much of which really centers around a definition of intelligence.) They have evolved over the 4.8 billion year history of the earth.

It is instructive to reflect on the way in which earth-based biological evolution must be done. Single cell entities arose out of the primordial soup roughly 2.8 billion years ago. A billion years passed before prokaryotic photos appeared. After almost another billion and a half years, around 180 million years ago, the first land and vertebrates arrived, and then humans 60 million years ago. Then things started moving fast. Reptiles arrived 270 million years ago, followed by dinosaurs at 200 and mammals at 180 million years ago. The first primates appeared 120 million years ago and the immediate predecessors to the great apes a mere 15 million years ago. Man arrived roughly 100 years from 2.5 million years ago. He learned agriculture a mere 10000 years ago, writing less than 8000 years ago and "expert" knowledge only came out in the last few hundred years.

This suggests that problem solving behavior, language, expert knowledge and application, reason, etc., are all pretty simple once the process of being and meaning are established. This means in the ability to move around in a dynamic environment, making the surroundings to a degree sufficient to achieve the necessary maintenance of life and reproduction. This part of intelligence is where evolution has concentrated its time—it is much harder.

[Dreyfus 71] has argued that human level intelligence was an extremely unlikely accident. The basis of his argument is that evolutionarily there is "no a priori advantage to intelligence, although it is a clear and unmistakable reproductive benefit" (Increased manipulation does not in itself lead to dominance, e.g. only two species of Proctosera remain and are survived by less-specialized but equally large competitors of similar ecological success—the basis is the length of time necessary devoted to searching of offspring during the full development of the nearly more complex central nervous system). He argues

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
 ARTIFICIAL INTELLIGENCE LABORATORY

A. I. Memo 809 May, 1988

ACHIEVING ARTIFICIAL INTELLIGENCE
 THROUGH BUILDING ROBOTS

Rodney A. Brooks

Abstract. We argue that generally accepted methodologies of Artificial Intelligence research are limited in the proportion of human level intelligence they can be expected to realize. We argue that the currently accepted decomposition and static representation used in such research are wrong. We argue for a shift to a process based model, with a decomposition based on task solving behavior as the organizational principle. In particular we discuss building robotic teams.

Acknowledgments. This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the research is provided in part by an IBM Faculty Development Award, in part by a grant from the Systems Development Foundation, and in part by the Advanced Research Projects Agency under Office of Naval Research contract N00014-86-K-0021.

© Massachusetts Institute of Technology 1986

AI through Building Robots

1. Introduction

There has been considerable philosophical debate on the possibility of "human level" artificial intelligence, centered around the notion that it requires as background the totality of previous which make up the human way of being in the world (Dreyfus 71, 81).

In this note we use a technical rather than philosophical argument that machines must indeed have a rich background of experience of being if they are to achieve human level intelligence. Unlike Dreyfus however, we conclude that (artificial) intelligent behavior is achievable with computers without the aid of heuristics, common-sense, or other holistic mechanisms. Rather, by adopting an incremental construction approach, progress towards this goal can be expected soon. (Naturally the author and his students are currently following this enlightened path.)

2. Three observations about the failure of AI.

In this section we give three reasons why AI has failed to live up to its early promise, and moreover why it will not live up to its current promise.

A. A lesson from evolution

We already have an extensive proof of the possibility of intelligent earlier-human beings. Additionally many animals are intelligent to some degree. (This is a subject of intense debate much of which really centers around a definition of intelligence.) They have evolved over the 4.8 billion year history of the earth.

It is instructive to reflect on the way in which earth-based biological evolution must be done. Single cell entities arose out of the primordial soup roughly 2.8 billion years ago. A billion years passed before prokaryotic photos appeared. After almost another billion and a half years, around 180 million years ago, the first land and vertebrates arrived, and then humans 60 million years ago. Then things started moving fast. Reptiles arrived 270 million years ago, followed by dinosaurs at 200 and mammals at 180 million years ago. The first primates appeared 120 million years ago and the immediate predecessors to the great apes a mere 15 million years ago. Man arrived roughly 100 years from 2.5 million years ago. He learned agriculture a mere 10000 years ago, writing less than 8000 years ago and "expert" knowledge only came out in the last few hundred years.

This suggests that problem solving behavior, language, expert knowledge and application, reason, etc., are all pretty simple once the process of being and meaning are established. This means in the ability to move around in a dynamic environment, making the surroundings to a degree sufficient to achieve the necessary maintenance of life and reproduction. This part of intelligence is where evolution has concentrated its time—it is much harder.

[Dreyfus 71] has argued that human level intelligence was an extremely unlikely accident. The basis of his argument is that evolutionarily there is "no a priori advantage to intelligence, although it is a clear and unmistakable reproductive benefit" (Increased manipulation does not in itself lead to dominance, e.g. only two species of Proctosera remain and are survived by less-specialized but equally large competitors of similar ecological success—the basis is the length of time necessary devoted to searching of offspring during the full development of the nearly more complex central nervous system). He argues

Figure 5. Test images: BROOKS 1 to 2 (top 2 images) and the reconstructed images. These reconstructed images at 87:1 compression are almost indistinguishable from the originals.

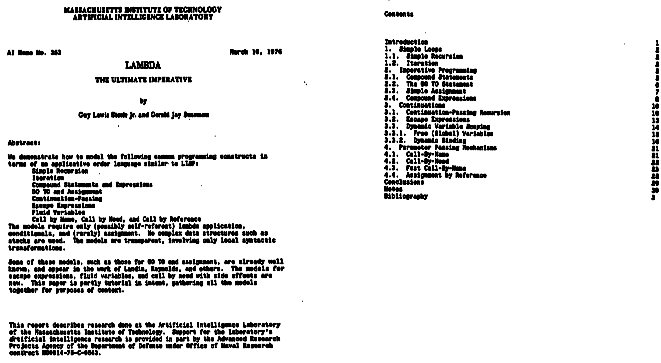
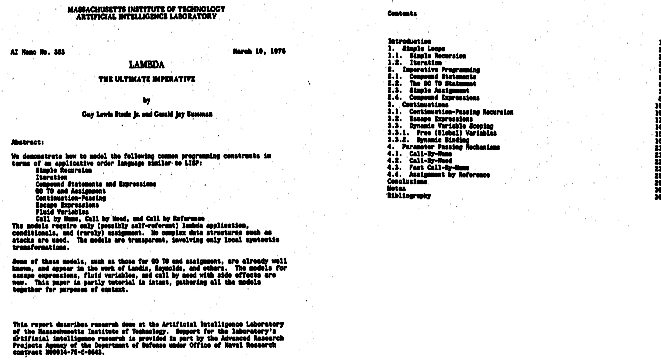


Figure 6. Test images: LAMBDA 1 to 2 (top 2 images) and the reconstructed images. The reconstructed images at 87:1 compression actually look better than the originals because isolated black pixels, an artifact of poor scanning, have been dropped.

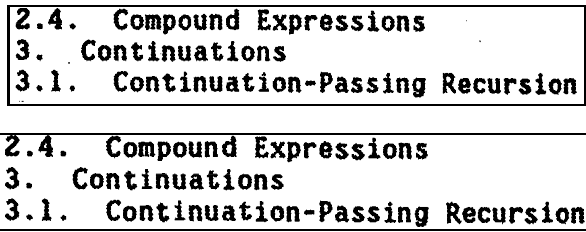


Figure 7. A zoomed portion of the original (up) and the reconstructed image of LAMBDA2.

Table 1. Compression of single-page documents

Test document	Compression ratio			
	CCITT Group 4	1-stage TIC	lossy CDIS	CDIS/TIC
CCITT1	28.4:1	77.7:1	95.9:1	80.4%
CCITT4	7.4:1	48.2:1	60.4:1	79.8%
BROOKS1	33.6:1	66.5:1	74.1:1	89.7%
BROOKS2	12.4:1	58.5:1	66.4:1	88.1%
BROOKS3	12.0:1	58.4:1	69.3:1	84.2%
BROOKS4	11.7:1	55.1:1	63.7:1	86.5%
LAMDA1	25.6:1	51.0:1	54.9:1	92.9%
LAMDA2	62.1:1	136.0:1	151.3:1	89.9%
LAMDA3	12.1:1	45.5:1	50.0:1	91.0%
Mean	15.3:1	60.0:1	69.0:1	86.9%

Table 2. A comparison of code space for pattern positions

Test document	Position bits		CDIS (lossless)		
	TIC	CDIS (lossless)	CDIS	/TIC	CDIS/TIC
CCITT1	9206	4068	2811	44.2%	30.5%
CCITT4	30942	15385	10308	49.7%	33.3%
BROOKS1	19934	12075	5325	60.6%	26.7%
BROOKS2	30270	20456	7221	67.6%	23.9%
BROOKS3	32597	21502	7477	66.0%	22.9%
BROOKS4	32654	21961	7603	67.3%	23.4%
LAMDA1	21950	10118	4424	46.1%	20.2%
LAMDA2	13012	5778	3738	44.4%	28.7%
LAMDA3	35032	21496	8287	61.4%	23.7%
Mean	25066	14760	6355	56.4%	25.9%

output is higher. As you can see, CDIS's output is about 82% of TIC's output.

Table 3. Compression of multi-page documents

Test document	Compressed file size		Compression ratio		
	TIC	CDIS	TIC	CDIS	CDIS/TIC
BROOKS	349,734	288,846	72.2:1	87.5:1	82.5%

Table 4 shows CDIS's output per character for CCITT1 and BROOKS. BROOKS uses only about two thirds as many bits per character as CCITT1 because the cost of transmitting each library pattern is amortized over a larger set of pattern instances. Various components of the output per character are also shown. CDIS compresses images to between 30 to 50 bits per character, an order of magnitude higher than the 2.2 bit per character lower bound (the entropy of English text). The pattern positions take up only about 3 bits per character, demonstrating the effectiveness of lossy position compression. The large number of pattern index bits (about 10 when we would have expected 2 or 3) indicates a shortcoming in pattern matching and library generation. Apparently, multiple versions of important characters are created in the library, inflating our bit counts and transmission cost. This is the most promising area for future work.

Table 4. Average bits per character for the single-page CCITT1 and the multi-page BROOKS

Test document	Characters	Library bits per character	Index bits per character	Position bits per character	Total bits per character
CCITT1	912	35.9	8.1	3.1	47.1
BROOKS	9043	18.3	11.0	2.6	31.9

6 DISCUSSION

CDIS runs on DEC Alpha workstations. Our current implementation uses some components (pattern extraction and the arithmetic coder) of the publicly available TIC system. The code has not yet been optimized for speed. On a 166 MHz Alpha, CDIS spends about 7 seconds compressing a 200 dpi letter size document, and about 1 second decompressing the document. This is much slower than CCITT compression. CDIS trades intensive computation for network bandwidth and disk space. This is the right tradeoff because computational power is doubling every two years while network bandwidth is increasing more slowly. For instance, private telephone bandwidth to rural areas is especially unlikely to improve in the near future.

The CDIS implementation demonstrates a wide margin of improvement in compression efficiency over the best previous lossy compression techniques. CDIS codes text positions by automatically formatting blocks of text, then transmitting positional errors for each pattern. These errors are small because CDIS rebuilds text source, and predicts pattern

positions accurately. The code size of the errors is cut further by coding in reduced precision.

The improved coding of pattern positions exposes the imperfection of pattern index compression. The cost per pattern index is well above the entropy per character of English text. This is because multiple versions of the same character are being added to the pattern library. A better pattern matching technique would result in a smaller library, which would lead to reduced costs for both the library itself and the pattern indices. To address this issue, we are currently working on pattern matching aspects of document image compression. We recently investigated an entropy based pattern matching algorithm derived from a scanner's registration based error model[12].

We are also working on a real time version of CDIS as part of a desktop sharing project.

REFERENCES

1. Kenneth R. McConnell, Dennis Bodson, and Richard Schaphorst, *FAX: Digital Facsimile Technology and Applications*, Artech House, 685 Canton Street, Norwood, MA 02062, second edition, 1992.
2. International Telegraph and Telephone Consultative 'Progressive bi-level image compression', in *Recommendation T.82*, (1993).
3. R. N. Ascher and G. Nagy, 'A means for achieving a high degree of compaction on scan-digitized printed text', in *IEEE Transactions on Computers*, c-23(11), pp. 1174–1179, (1974).
4. M. J. Holt and C. S. Xydeas, 'Recent developments in image data compression for digital facsimile', *ICL Technical Journal*, 123–146, (1986).
5. K. M. Mohiuddin, *Pattern matching with application to binary image compression*, Ph.D. dissertation, Stanford University, Stanford, CA, 1982.
6. I. H. Witten, T. C. Bell, H. Emberson, and S. Inglis, 'Textual image compression: two-stage lossy/lossless encoding of textual images', *Proc. IEEE*, **86**(6), 878–888, (1994).
7. K. M. Mohiuddin, 'Lossless binary image compression based on pattern matching', in *Proc. International Conference on Computers, System and Signal Processing*, pp. 447–451, (1984).
8. W. K. Pratt, P. J. Captitant, W. H. Chen, E. R. Hamilton, and R.H. Wallis, 'Combing symbol matching facsimile data compression system', *Proc. IEEE*, **68**(7), 786–796, (1980).
9. O. Johnsen, J. Segen, and G. L. Cash, 'Coding of two-level pictures by pattern matching and substitution', *Bell Systems Technical Journal*, **62**(8), 2513–2545, (1983).
10. Timonhy C. Bell, John G. Cleary, and Ian H. Witten, *Text Compression*, Prentice Hall, Englewood Cliffs, NJ, 1990.
11. I. H. Witten, Moffat A, and T. C. Bell, *Managing Gigabytes: compressing and indexing documents and images*, Van Nostrand Reinhold, New York, 1994.
12. Qin Zhang and John M. Danskin, 'Entropy-based template matching for document image compression'. accepted by ICIP, 1996.