Memory-Efficient Algorithms for Raster Document Image Compression*

Maribel Figuera

School of Electrical & Computer Engineering

Ph.D. Final Examination

June 13, 2008

Committee Members:

Prof. Charles A. Bouman (Chair)
Prof. Jan P. Allebach
Prof. George Chiu
Prof. Edward J. Delp
Prof. James V. Krogmeier

*Research sponsored by Samsung Electronics and Hewlett-Packard

PURDUE UNIVERSITY

Outline

- Motivation for Research
- Cache-Efficient Dynamic Dictionary Design for Multi-Page Document Compression with JBIG2*
  - Mixed Raster Content and JBIG2
  - Research Goals
  - Review of Prior Work
  - Contributions
    - » Dynamic Symbol Caching Algorithm
    - » Prescreened Weighted-XOR (PWXOR) Pattern Matching Criterion
    - » Adaptive Striping
  - Experimental Results
  - Conclusion
- Hardware-Friendly Mixed Content Compression Algorithm**
  - Mixed Content Compression Algorithm
  - Experimental Results
  - Conclusion

*Sponsored by Samsung Electronics Corporation
**Sponsored by Hewlett-Packard Company

Motivation for Research

- Raster scanned representations of documents are required by many document imaging applications:
  - scan-to-print, scan-to-email, print spooling, archiving or document exchange
- Documents in raster format can be quite large uncompressed:
  - 96.3 MB for a full-color 600 dpi letter-sized document
- Effective raster document compression is very important
- Typical low bit rate compression formats:
  - Mixed Raster Content (MRC)
    - \+ Maintains text/graphics quality by using a binary mask layer to encode the high-resolution transitions
    - \- Requires large memory buffers
  - JPEG
    - \+ Small memory buffers
    - \+ Suitable for photographic images or continuous tone images
    - \- Presents distortion on sharp edges (e.g. in text/graphics)

Mixed Raster Content (MRC) and JBIG2



INPUT DOCUMENT IMAGE

RESTORED DOCUMENT IMAGE

M * FG + M' * BG



FOREGROUND

BINARY MASK

BACKGROUND

JPEG / JPEG2K

JBIG1 / JBIG2

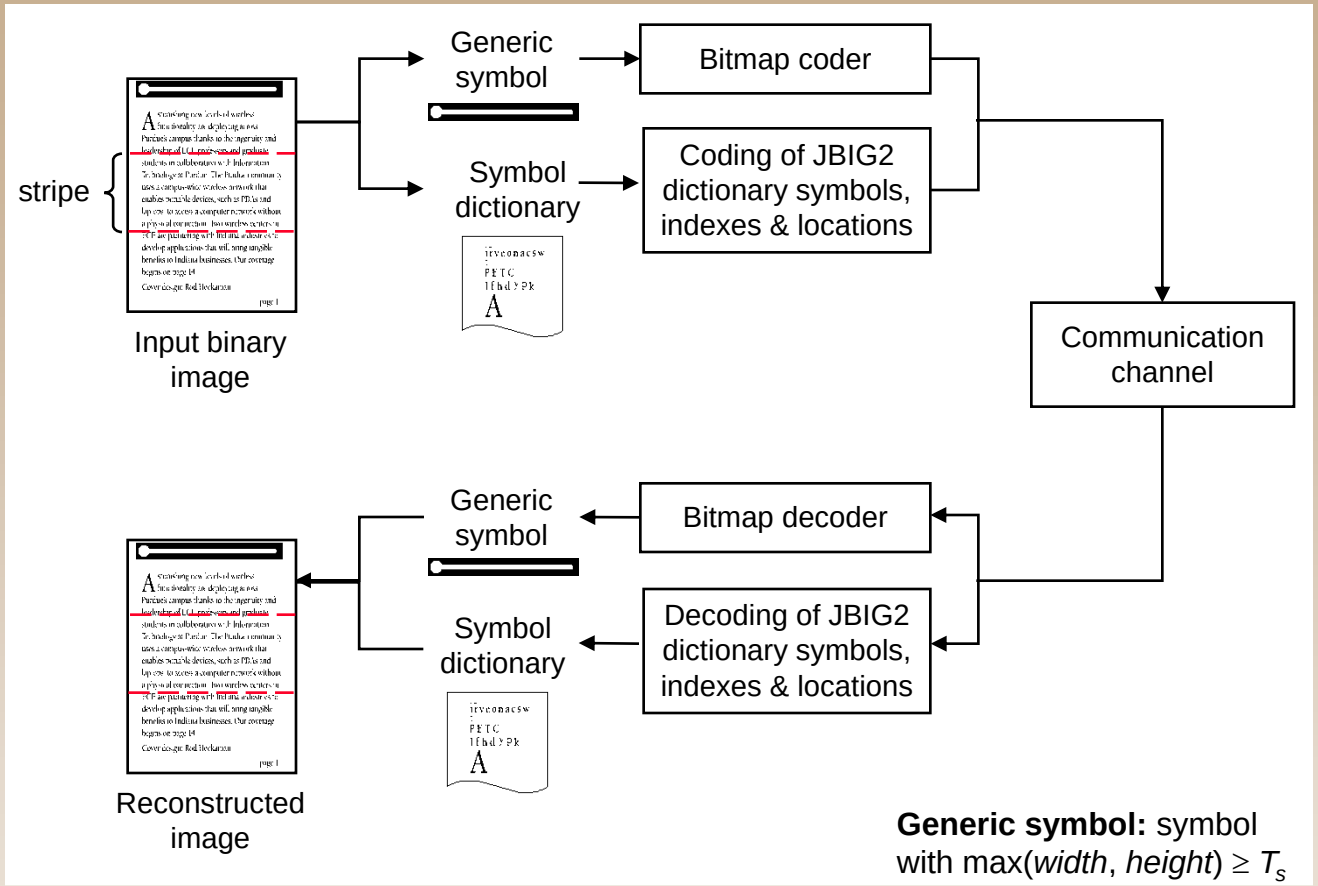
JPEG / JPEG2K

LAYER SEGMENTATION

COMPRESSION METHOD

PURDUE UNIVERSITY

JBIG2 Encoding System



What is important:

- Dictionary design
- Symbol matching criterion

PURDUE UNIVERSITY

Research Goals

- Introduce a novel dynamic symbol caching dictionary
  - Reduces bit rates for multi-page documents
  - Reduces memory requirements for JBIG2 encoder/decoder
  - Allows for more fine grain striping of page
- Introduce a fast pattern matching technique (PWXOR)
  - Reduces pattern matching time
  - Achieves high accuracy and low bit rates
- Introduce an adaptive striping method for page striping
  - Reduces bit rates relative to fixed striping
- Experimental results
  - Obtain multi-page test documents with variety of content
  - Benchmark with existing JBIG2 commercial encoders
  - Implementation of JBIG2 standard and verification

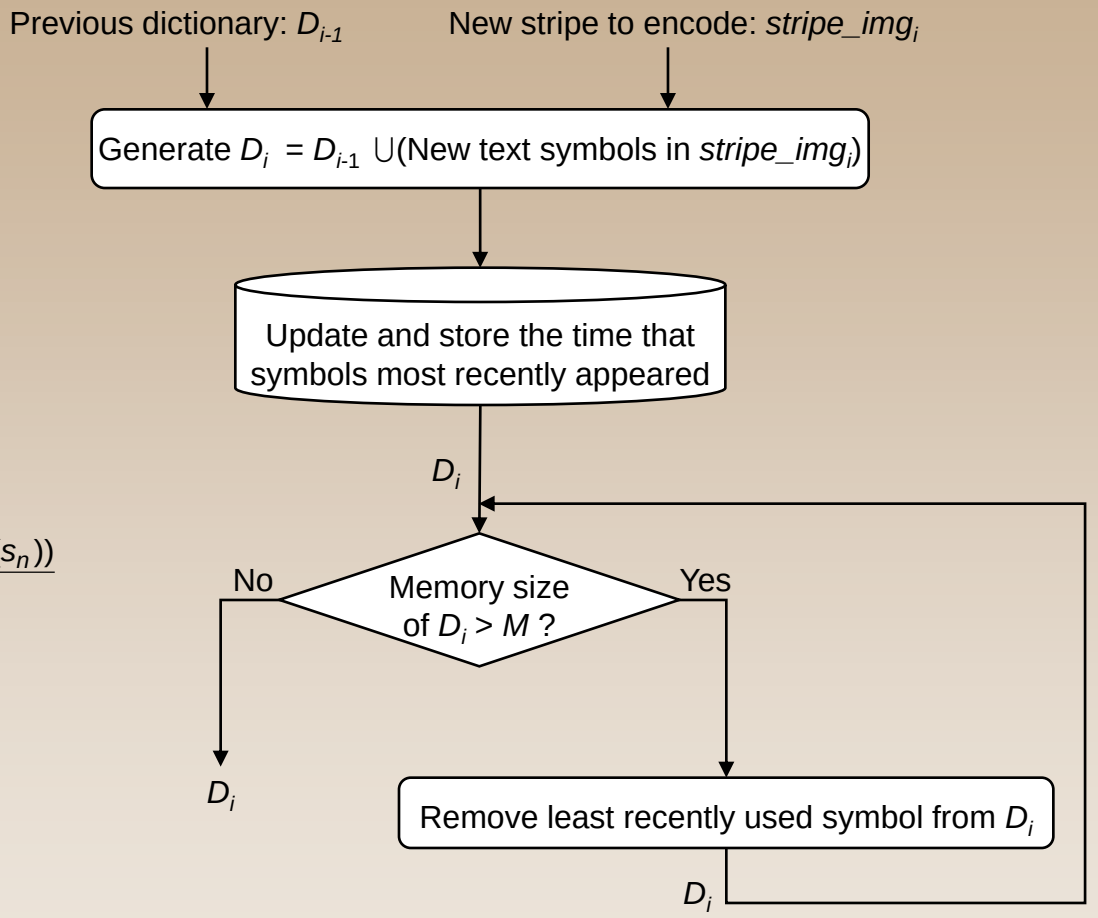
PURDUE UNIVERSITY

Summary of Symbol Dictionaries Attributes

		Update	Dictionary Size	Raster Buffer Size	Bit Rate	Coding Time
Prior Dictionaries	**Independent Static**	New for each stripe	Small	Single stripe	High	Low
	Local Dynamic	Recursive	Small	Single stripe	Medium	Low
Proposed Dictionaries	**Global**	Static	Unlimited	Entire multi-page document	Low	High
	Optimal Dynamic	Recursive	Unlimited	Entire multi-page document	Low	High
	Dynamic Symbol Caching	Recursive	User selectable	Single stripe	Low	Medium

PURDUE UNIVERSITY

Dynamic Symbol Caching Algorithm



*Memory Size of Dictionary D_i :

$$MSD_i = \sum_{n=1}^{N} \frac{32 + \text{round}(W(s_n) \times H(s_n))}{8}$$

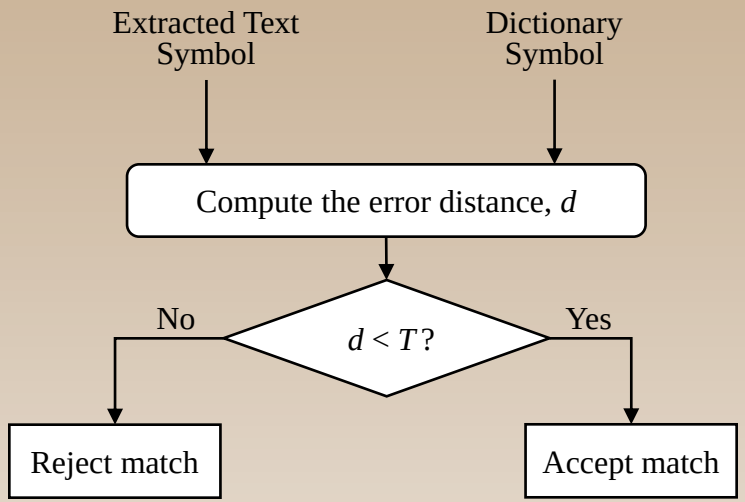
$M \leq 1$ Mbyte

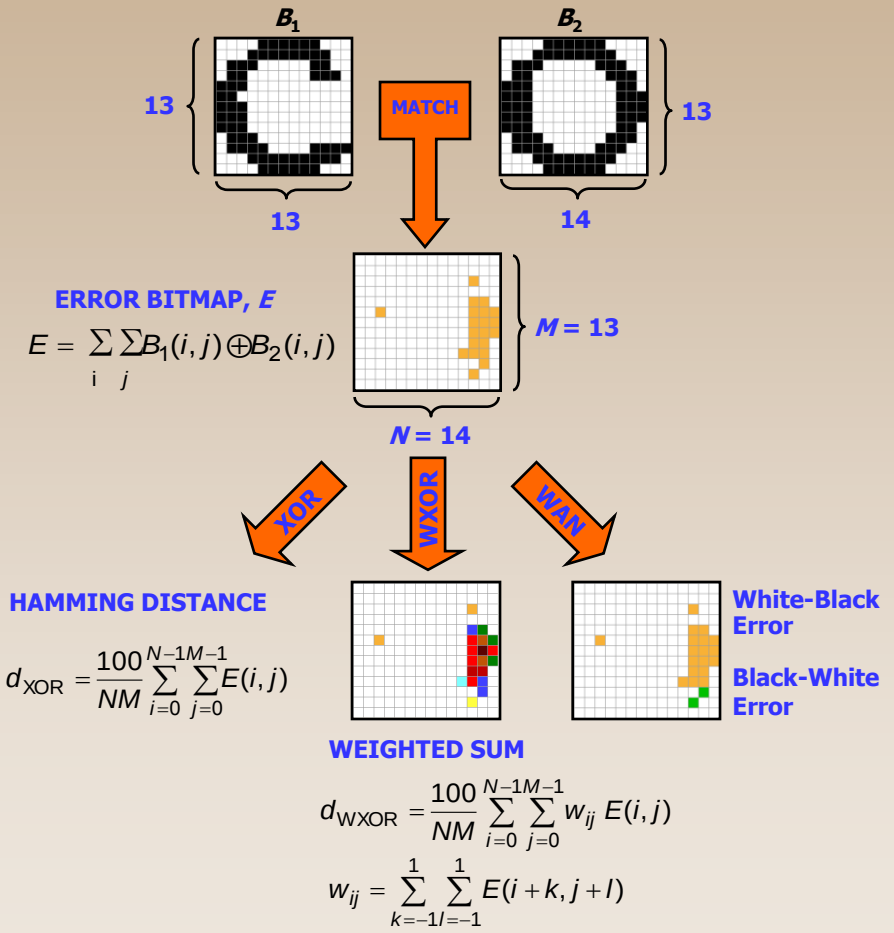
$$D_i = \begin{cases} \bigcup_{n=1}^{i} S_n & \text{if } MSD_i \leq M \\ \bigcup_{n=1}^{i} S_n \setminus \tilde{S}_m & \text{if } MSD_i > M \end{cases}$$

PURDUE UNIVERSITY

Typical Pattern Matching Criteria

- Compute a distance metric, *d*, based on an error map
- Each matching threshold, *T*, must be chosen to avoid substitution errors (hb; ceo; LD)

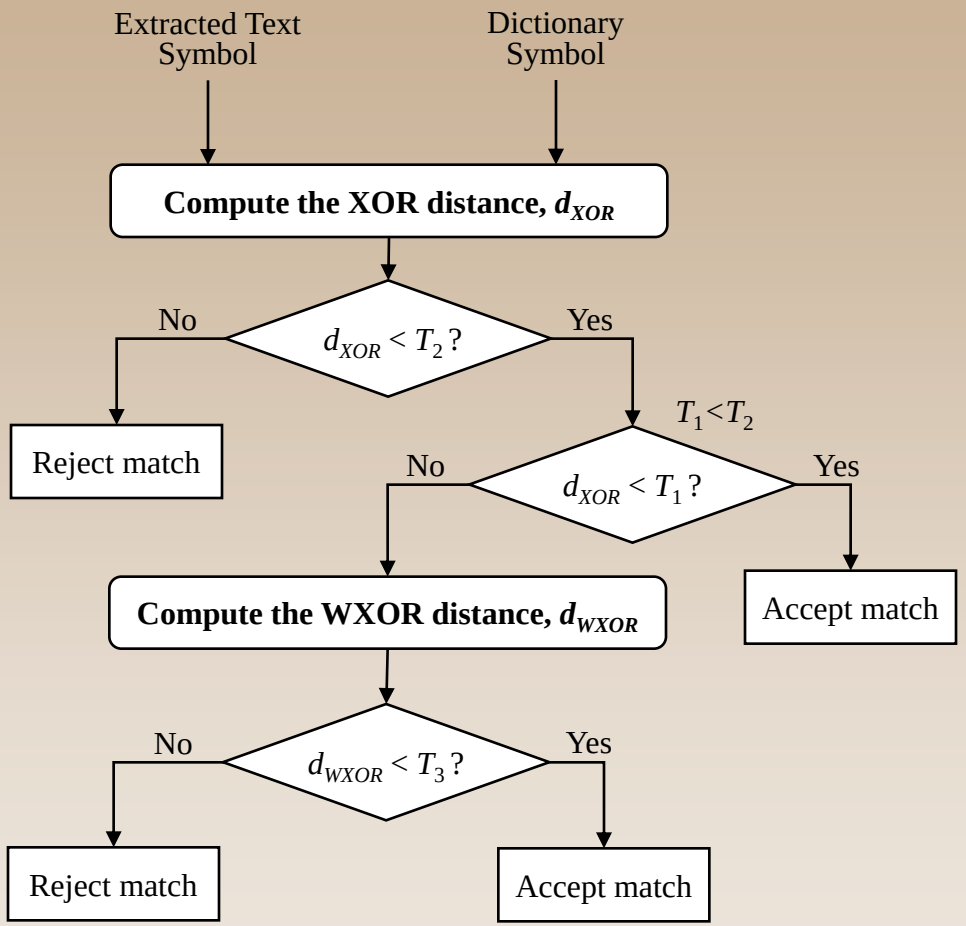




Criterion	Complexity	Bit rate
XOR (Exclusive-OR)	Low	High
WXOR (Weighted XOR)	High	Low
WAN (Weighted AND NOT)	High	Low

PURDUE UNIVERSITY

New Pattern Matching Criterion: PWXOR



PURDUE UNIVERSITY

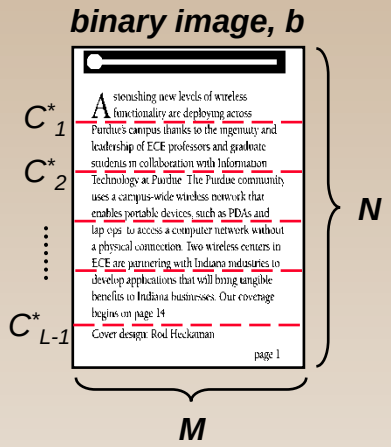
Adaptive Striping

- **Page Striping:**
  - Split multi-page document into ***L*** horizontal stripes per page
  - Pros: Allows for smaller encoder/decoder raster buffer sizes
  - Cons: May cause symbols to be split → larger dictionaries → higher bit rates
- **Goal:** Minimize # of split symbols
- **Approach:** Adapt each stripe height to the content of the page

  $C_k(i)$: # of black-to-white transitions in line i

$$C_k^* = \min\{C_k(i)\}$$

$$C_k(i) = \sum_{j=1}^{M-1} b(i, j-1)\left[1 - b(i,j)\right], \quad i \in \left\{k\left\lfloor\frac{N}{L}\right\rfloor - 25, k\left\lfloor\frac{N}{L}\right\rfloor + 25\right\}$$



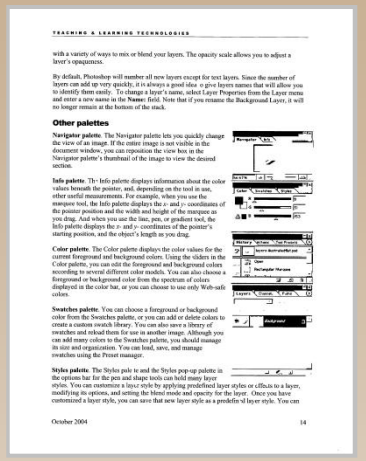
Striping method	Location of stripe break points	Bit rate	Dictionary size	Complexity
Fixed (prior method)	$k\lfloor N/L \rfloor$, $1 \le k \le L\text{-}1$	High	Large	Low
Adaptive (our new method)	C^*_k, $1 \le k \le L\text{-}1$	Low	Small	Low

PURDUE UNIVERSITY

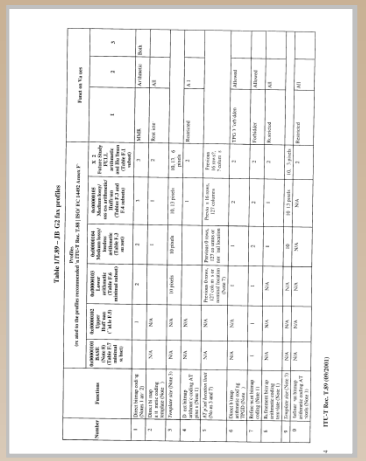
Sample Test Images
(Full test suite: 162 pages; 300 dpi)



Wavelinks Magazine
(23 pages)



Photoshop CS Manual
(46 pages)



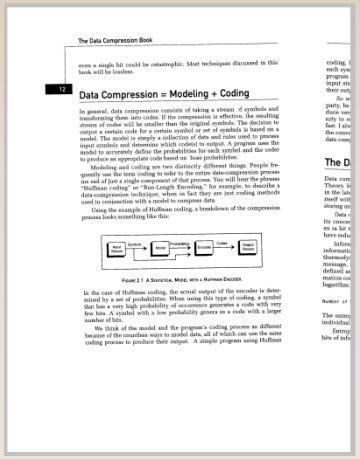
T.89 Standard
(18 pages)



Time Magazine
(28 pages)



ECE Impact Magazine
(28 pages)



Data Compression Book
(19 pages)

PURDUE UNIVERSITY

Comparison of XOR, WXOR, and PWXOR

Test Image
- 2560 x 3280
- 3971 symbols
- 300 dpi



Thresholds

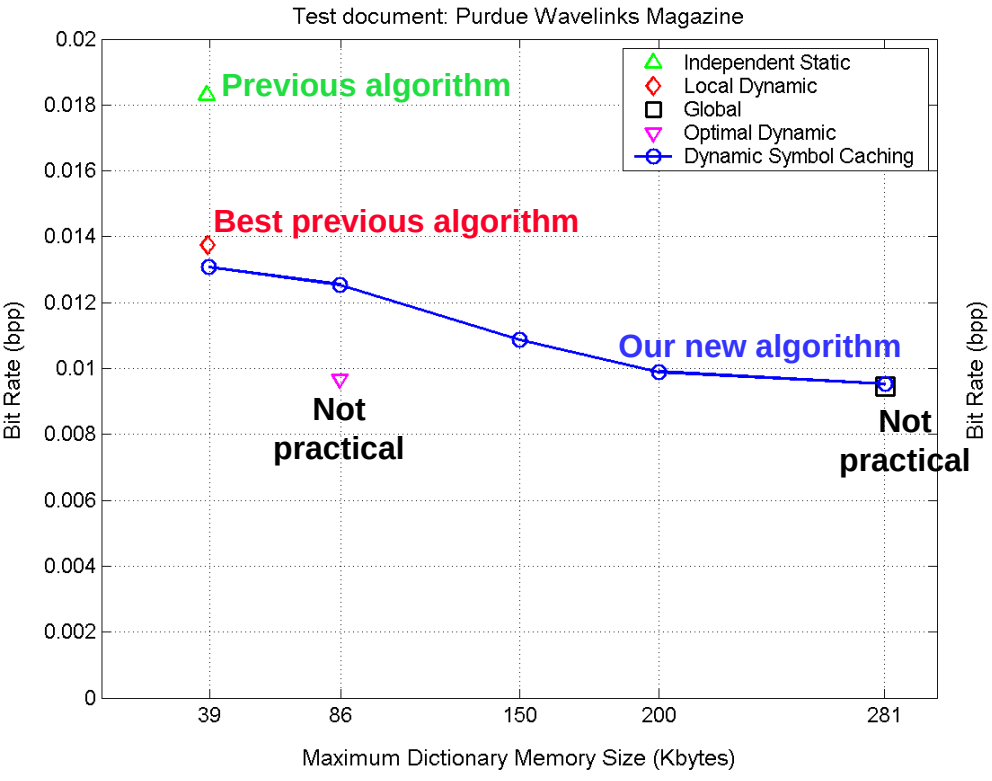
	T_s	T_1	T_2	T_3
XOR	600	4	-	-
WXOR	600	-	-	23
PWXOR	600	4	14	23

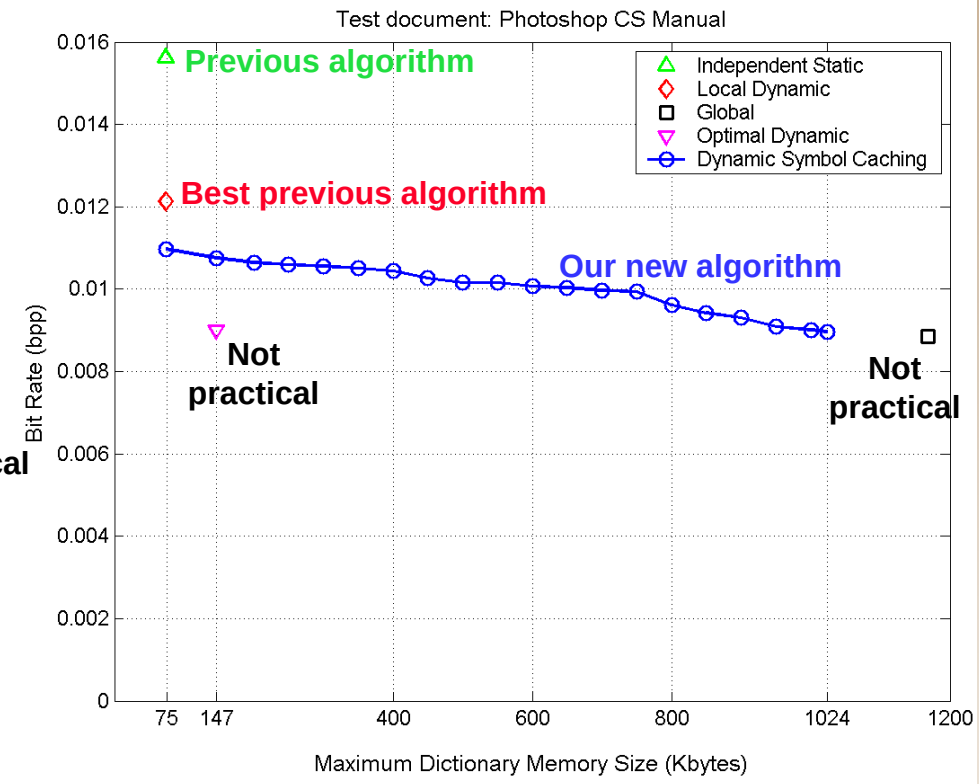
Matching Criterion	Bit rate (bpp)	Dictionary size (# of symbols)	Pattern matching time (seconds)	Average # of matching operations	
				XOR	WXOR
XOR	0.021410	531	0.81	21.57	0
WXOR	0.016386	310	1.56	0	9.83
PWXOR (new criterion)	0.016401	310	0.55	9.85	1.64

- Fastest criterion in finding a match
- Nearly same bit rate as that of WXOR
- Reduces WXOR operations by 83%

PURDUE UNIVERSITY

Bit Rate Comparison for 2 stripes/pg



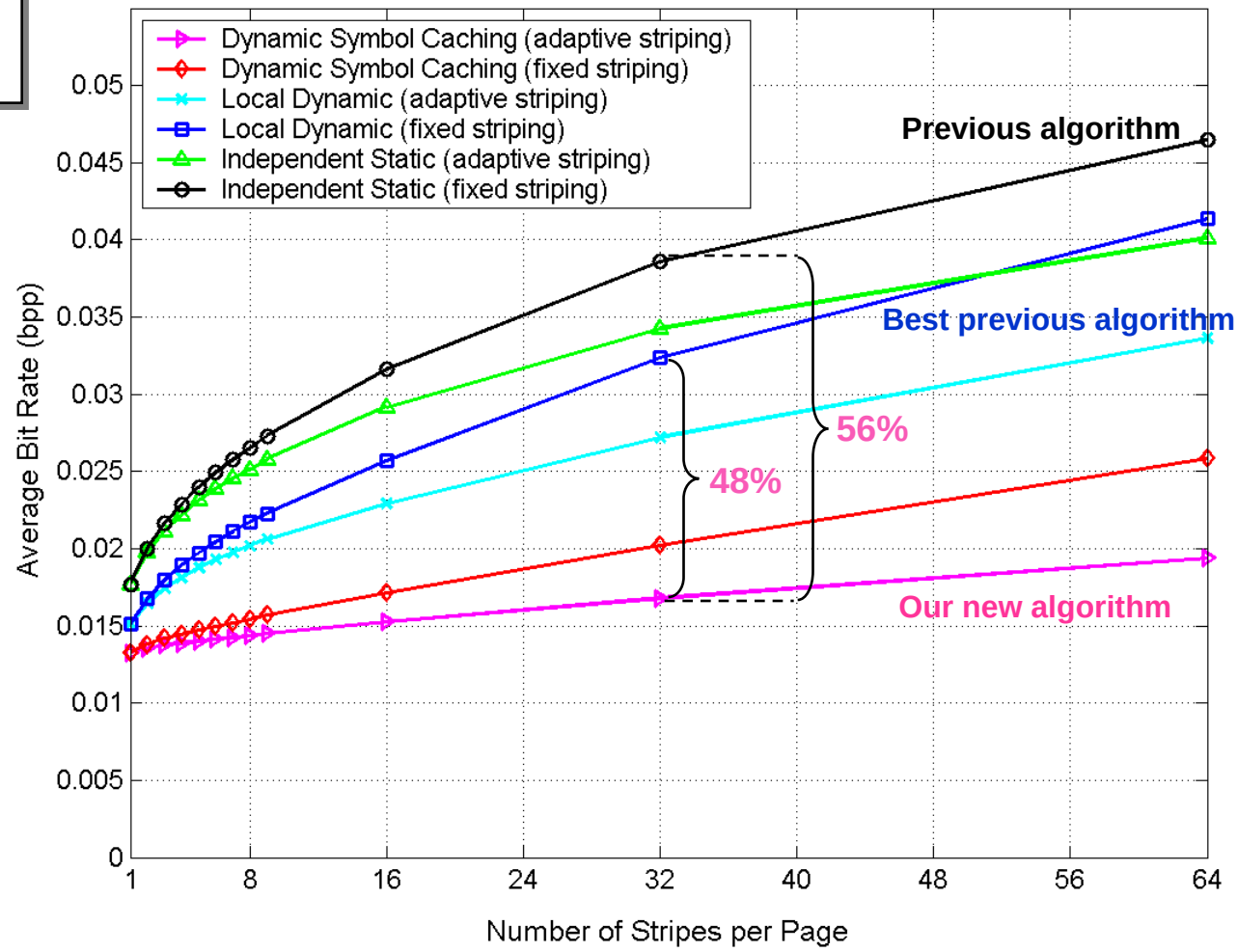


PURDUE UNIVERSITY

Adaptive vs. Fixed Striping

Test Document

- 6 multi-page documents (162 pages)
- resolution 300 dpi



PURDUE UNIVERSITY

Percentage of Average Bit Rate Reduction achieved by Dynamic Symbol Caching with Adaptive Striping relative to Previous Methods

- Test document: 6 multi-page documents (162 pages)

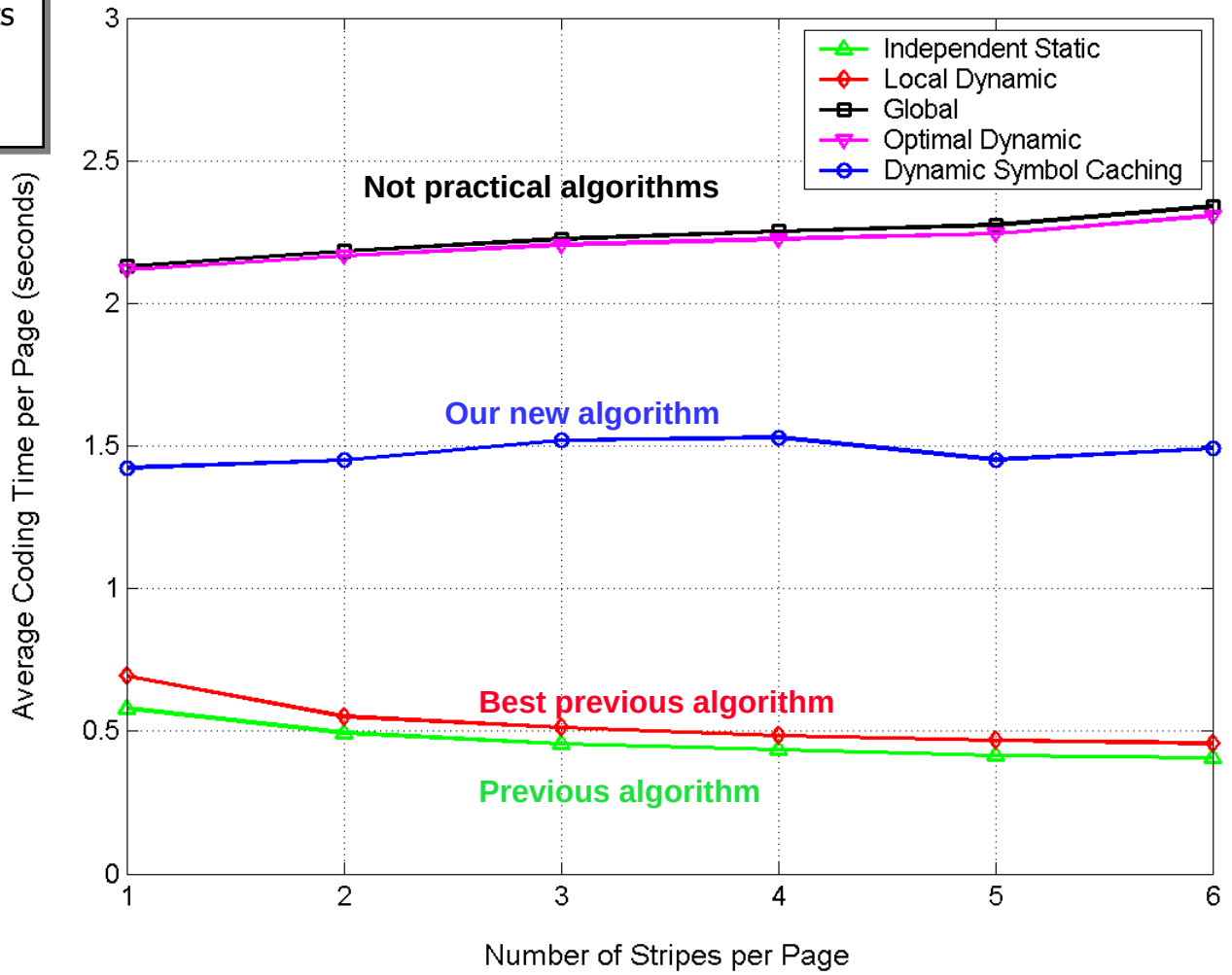
Stripes per page	1	2	3	4	6	8	16	32	64
Bit rate reduction of Dynamic Symbol Caching (adaptive) relative to Dynamic Symbol Caching (fixed)	0%	2%	3%	4%	6%	7%	11%	17%	25%
Bit rate reduction of Dynamic Symbol Caching (adaptive) relative to Independent Static	25%	32%	36%	39%	43%	46%	52%	56%	58%
Bit rate reduction of Dynamic Symbol Caching (adaptive) relative to Local Dynamic (best previous method)	12%	19%	23%	27%	31%	34%	41%	48%	53%

PURDUE UNIVERSITY

Average Encoding Time as a Function of the Number of Stripes per Page

Test Document

- 6 multi-page documents (162 pages)
- resolution 300 dpi



PURDUE UNIVERSITY

Comparison to Commercial JBIG2 Encoders

- Test document: 6 multi-page documents (162 pages)
- *Encoder parameters chosen to produce lowest bit rates with no visible substitution errors

Encoder	Entropy/Bitmap coding method	Bit rate (bpp)	Compression ratio average
PJBIG2* (our encoder)	Huffman/MMR	0.01327	75:1
AccuSoft ImageGear	Huffman/MMR	0.01597	63:1
Pegasus Imaging*	Huffman/MMR	0.01842	54:1
CVista PdfCompressor*	Huffman/MMR	0.01858	54:1
Pegasus Imaging*	Arithmetic (MQ coder)	0.01425	70:1
CVista PdfCompressor*	Arithmetic (MQ coder)	0.01465	68:1
Luratech LuraDocument PDF	Arithmetic (MQ coder)	0.02344	43:1
Snowbound Snowbatch	Arithmetic (MQ coder)	0.03397	29:1

PURDUE UNIVERSITY

Conclusion

- Proposed novel Dynamic Symbol Caching algorithm
  - Reduces bit rates by between 12% and 53% relative to the best previous dictionary design
  - Provides for efficient encoding of binary documents with smaller encoder and decoder buffers than are often used
  - Achieves lowest bit rates among all the JBIG2 commercial encoders
- Introduced adaptive striping
  - Minimizes the number of split symbols
  - Generates smaller symbol dictionaries
  - Reduces bit rates by between 2% and 25%
- Presented a fast pattern matching algorithm: PWXOR
  - Reduces bit rate by 23% relative to that of XOR, while requiring only 68% of the matching time of XOR
  - Achieves very nearly the same bit rate as WXOR, but only requires 35% of the matching time of WXOR

Related Patents and Publications

- “Apparatus and method of dynamically caching symbols to manage a dictionary in a text image coding and decoding system”. Jonghyon Yi, Hyung-Soo Ohk, Charles A. Bouman, and Maribel Figuera; United States Patent submitted January 2007, filed January 2008. (Licensed to Samsung Electronics Corporation)

- “Apparatus and method of matching symbols in a text image coding and decoding System”. Jonghyon Yi, Hyung-Soo Ohk, Charles A. Bouman, and Maribel Figuera; United States Patent submitted January 2007, filed January 2008. (Licensed to Samsung Electronics Corporation)

- “A new approach to JBIG2 binary image compression”. Maribel Figuera, Jonghyon Yi, and Charles A. Bouman; *IS&T/SPIE Conference on Color Imaging XII: Processing, Hardcopy, and Applications*; Proceedings SPIE Volume 6493, 649305; San Jose, California, January 2007

- “Cache-efficient dynamic dictionary design for multi-page document compression with JBIG2”. Maribel Figuera, Jonghyon Yi, and Charles A. Bouman; submitted to *The Journal of Electronic Imaging*, June 2008

Outline

- Motivation for Research
- Cache-Efficient Dynamic Dictionary Design for Multi-Page Document Compression with JBIG2*
  - Mixed Raster Content and JBIG2
  - Research Goals
  - Review of Prior Work
  - Contributions
    - » Dynamic Symbol Caching Algorithm
    - » Prescreened Weighted-XOR (PWXOR) Pattern Matching Criterion
    - » Adaptive Striping
  - Experimental Results
  - Conclusion
- Hardware-Friendly Mixed Content Compression Algorithm**
  - Mixed Content Compression Algorithm
  - Experimental Results
  - Conclusion

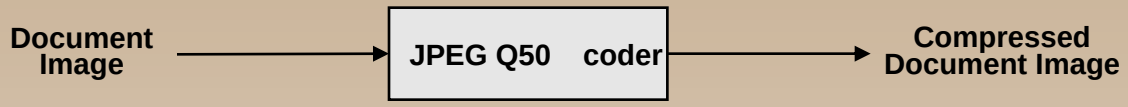
*Sponsored by Samsung Electronics Corporation
**Sponsored by Hewlett-Packard Company

Mixed Content Compression (MCC)

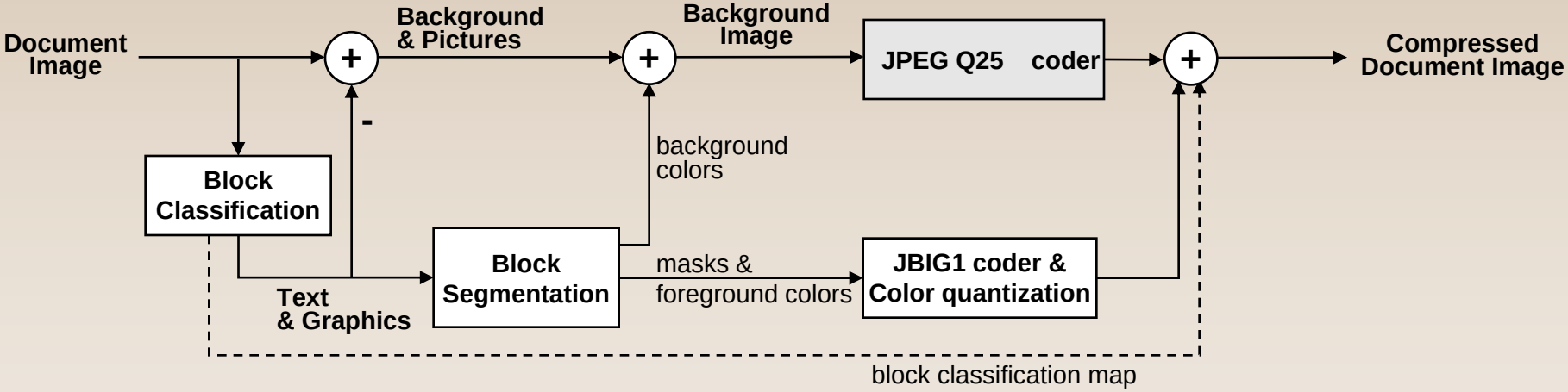
- Algorithm design based on 300 dpi scans from HP low cost all-in-ones
- Hardware efficient architecture by using 8 row buffer of pixels
- Block-based and content-adaptive
  - Divides image into 8x16 pixels blocks
  - Classifies each block into background/picture block or *n*-color block (*n* = 2, 3, 4)
  - Uses different encoding methods specifically designed for each block class
  - Includes a JPEG coder and a JBIG1 coder
- Comparison:
  - Baseline A - JPEG at Q50
    - 0.55 bpp (44:1 compression)
    - Sufficient image quality
  - Baseline B - JPEG at Q25
    - 0.39 bpp (62:1 compression)
    - Unacceptable text quality
  - Mixed Content Compression (MCC) using JPEG at Q25
    - 0.34 bpp (71:1 compression)
    - Sufficient image quality

General Structure of MCC vs. JPEG

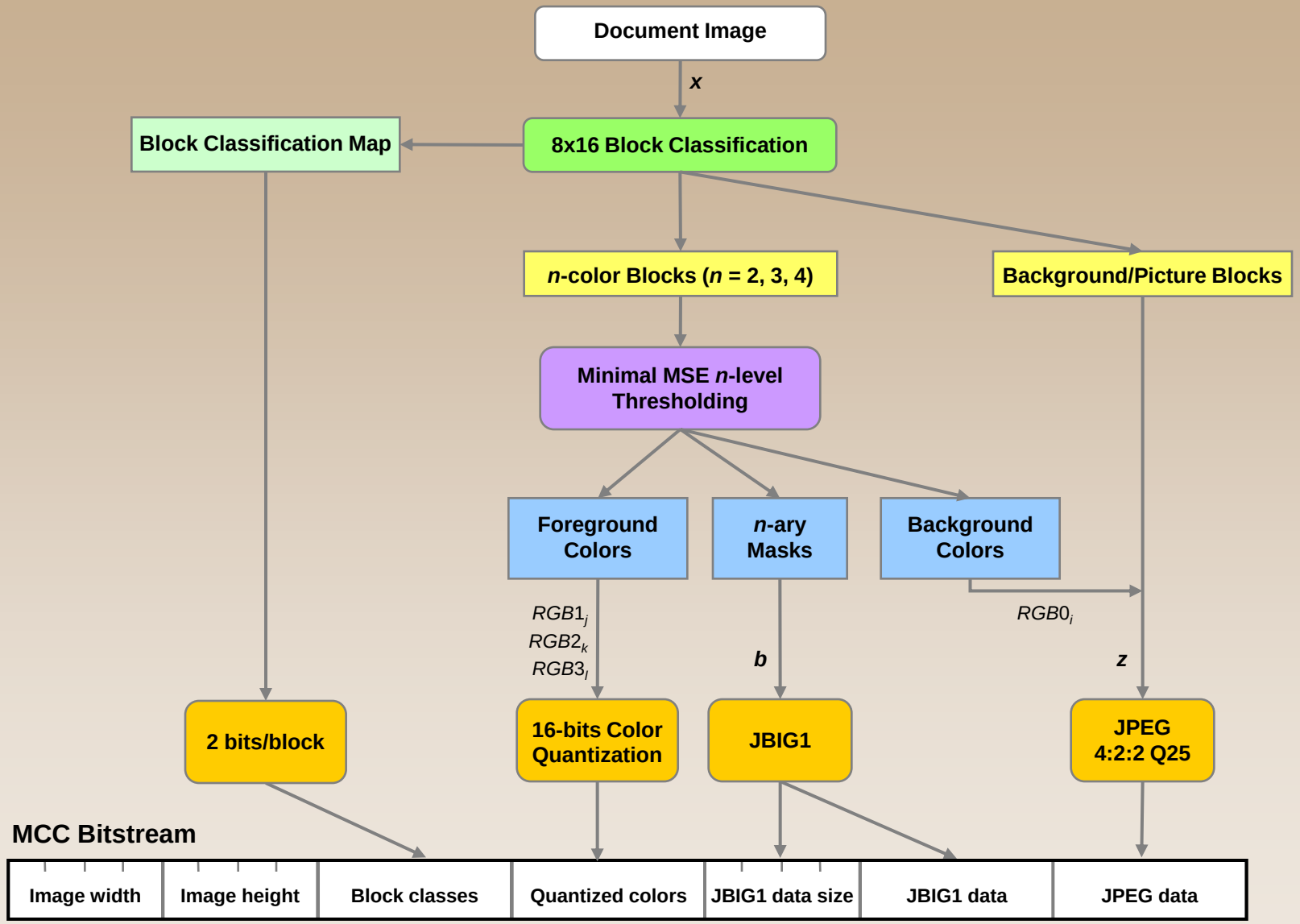
JPEG Structure:



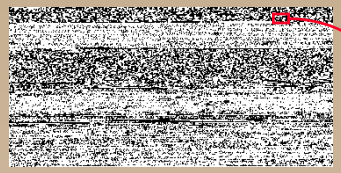
MCC Structure:



MCC Encoder Flow Diagram



MCC Encoding Example



Portion of *b*



40

58

Binary masks

b

JBIG1 coder

n-color blocks

Foreground colors

16-bits Color Quantization

Background colors



Original image, *x*

Fill-in blocks

Copy blocks

MCC bitstream

Background/picture blocks

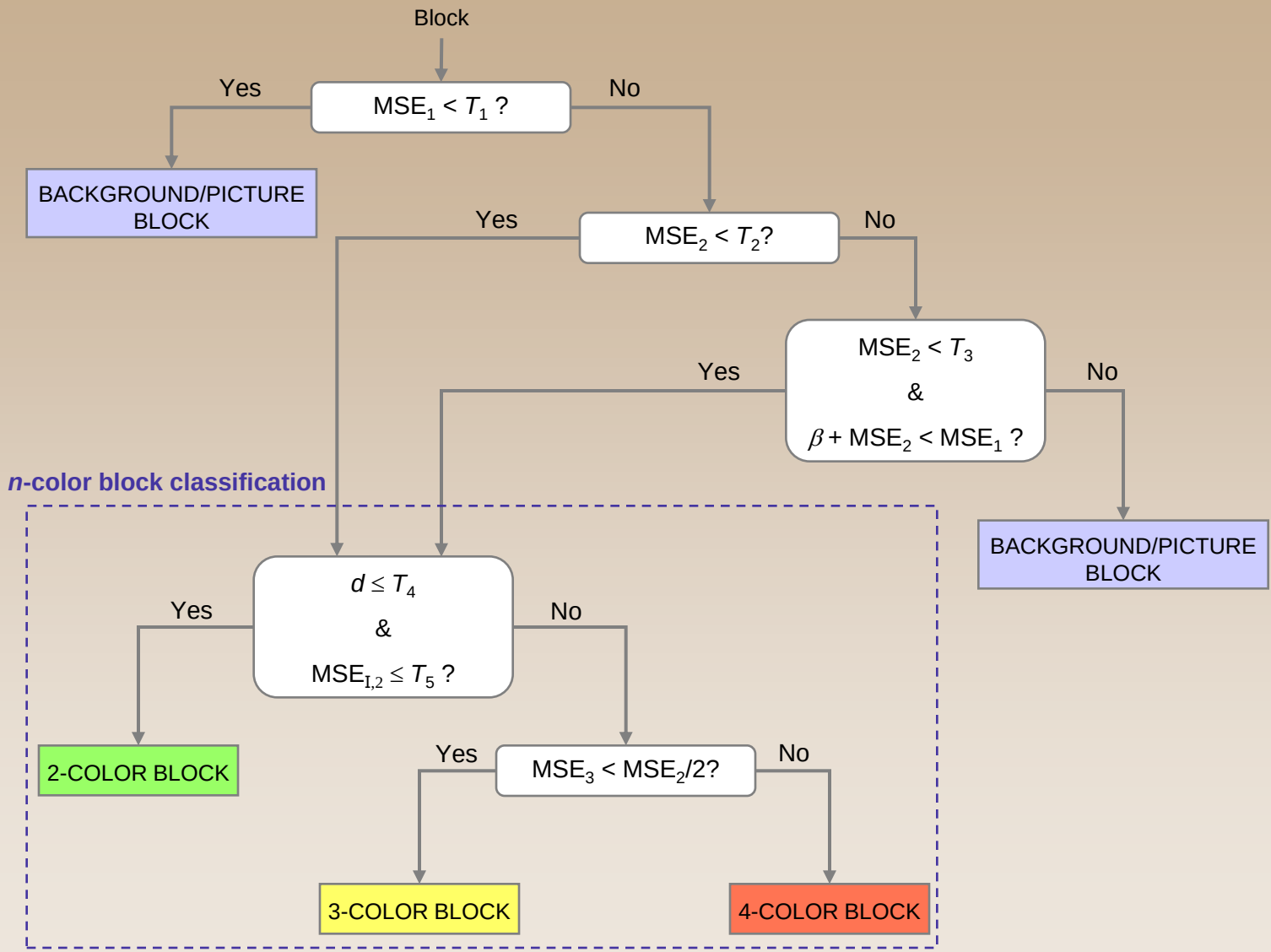


Background image, *z*

JPEG Q25 coder

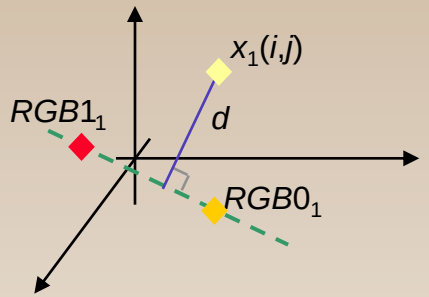
Block classification

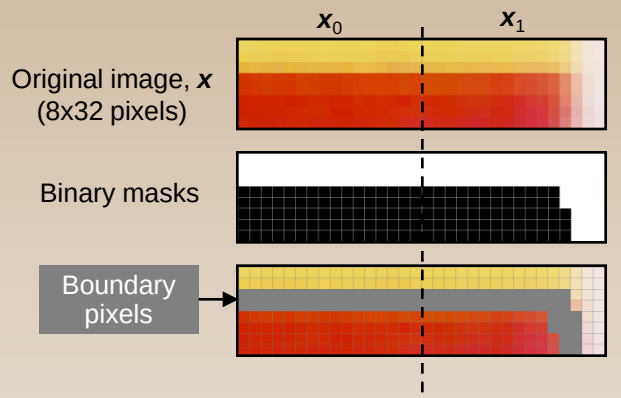
MCC Block Classification Algorithm



n-Color Block Classification

- For a block, $\boldsymbol{x}_k$, to be represented accurately by $n = 2$ colors, two conditions must be met
  - The average distance, d, of boundary pixels to line determined by median colors $RGB1_k$ and $RGB2_k$ must be $\leq T_4$
    - » Use binary mask of 2-color block to find boundary pixels, $x_k(i, j)$, in block



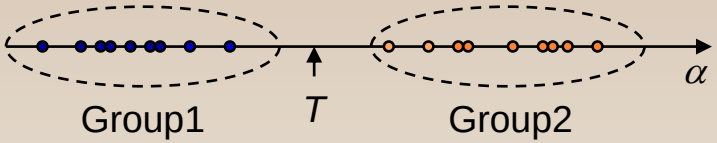


  - The MSE between interior pixels in the original block and its 2 color representation must be $\leq T_5$
- If block requires $n > 2$ colors. Then, compute MSE of block when it is represented by 3 colors, MSE_3
  - If $MSE_3 < MSE_2/2$, block is 3-color. Otherwise, block is 4-color

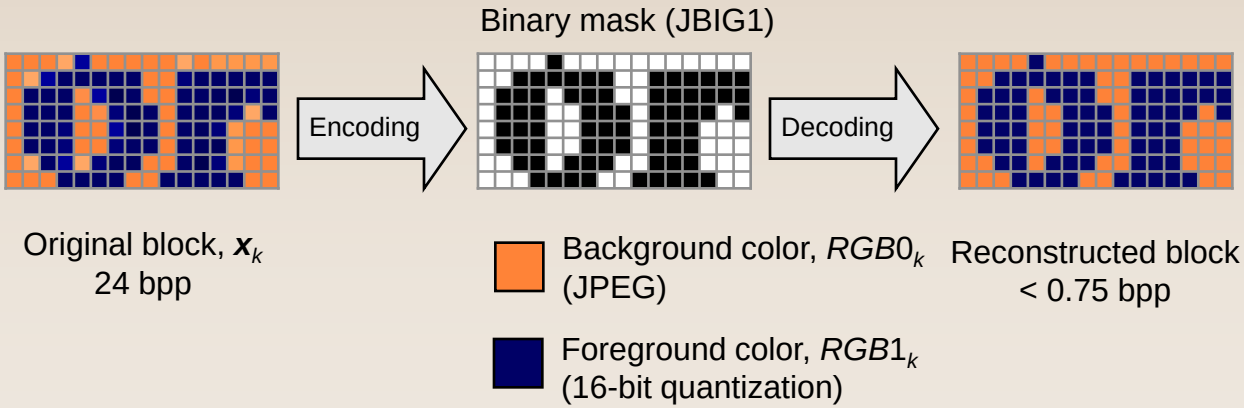
Segmentation of a 2-Color Block

Minimal MSE bi-level thresholding

- Choose the color axis, α, with the largest variance among the three color axes (R, G, or B)
- A threshold, *T,* on the chosen color axis partitions 8x16 pixels into two groups
- Select threshold *T* that minimizes the MSE γ_α^2



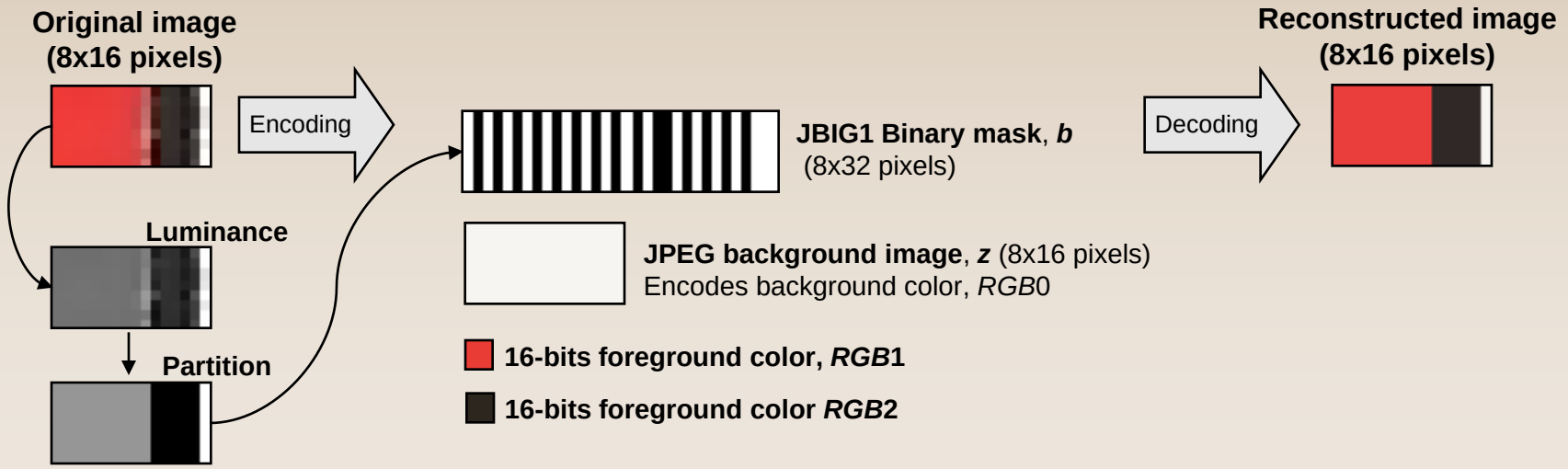
$$\gamma_\alpha^2 = \frac{N_{\alpha,0}\,\sigma_{\alpha,0}^2 + N_{\alpha,1}\,\sigma_{\alpha,1}^2}{N_{\alpha,0} + N_{\alpha,1}}$$



Segmentation of a 3 or 4-Color Block

Minimal MSE *n*-level thresholding (*n* = 3, 4)

- Compute luminance of block
- Partition luminance into *n* groups of pixels
  - » Choose luminance partition that minimizes MSE(R, G, B)
- Compute *n*-color values as the average of pixels in each group. Use only interior pixels unless all pixels in the group are boundary pixels
- Convert *n*-value mask to binary mask



Experimental Results

- Encoders settings

Encoder	Block classification parameters						JPEG settings		JBIG1 settings	
	T_1	T_2	T_3	T_4	T_5	β	Quality level	Subsampling mode	Compression mode	Lines per stripe
MCC	200	200	4000	20	200	500	25	4:2:2	sequential	8
JPEG	–	–	–	–	–	–	50	4:2:2	–	–

- Image database
  - 600 images scanned at 300 dpi and 24 bpp
  - All scanned images were descreened to remove halftone noise
  - Six image categories, composed of 100 images each
    - mono mixed
    - color mixed
    - mono text
    - color text
    - mono photo/picture
    - color photo/picture

Sample Descreened Test Images

(Full test suite: 600 images)



colormix300dpi



colormix110



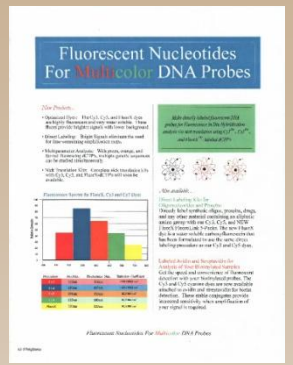
colormix114



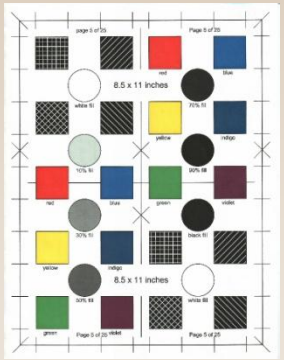
colormix117



monomix9



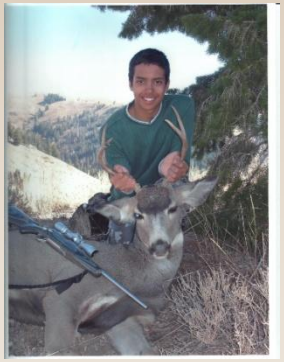
colortext6



colortext16



monotext12



colorpic11



monopic16



colorphoto6



monophoto32

Sample Block Classifications*

2-color
3-color
4-color
Background/Picture

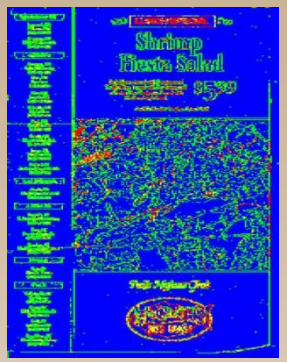


colormix300dpi

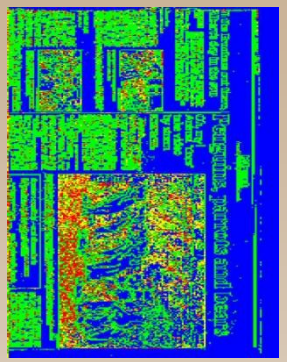


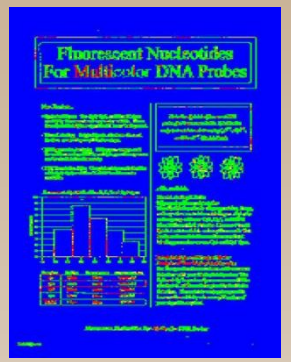
colormix110


colormix114

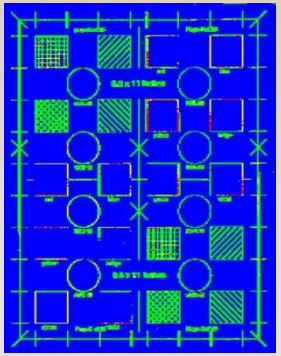


colormix117


monomix9

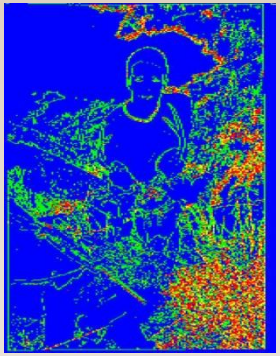


colortext6

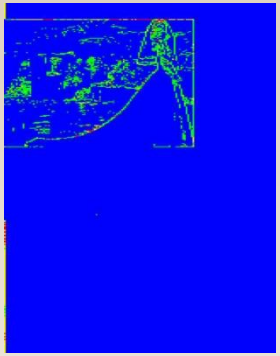

colortext16

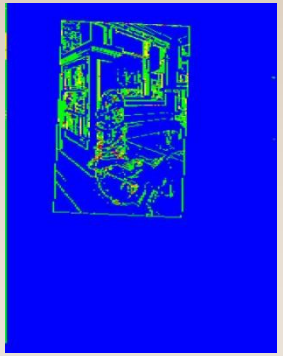


monotext12


colorpic11


monopic16


colorphoto6

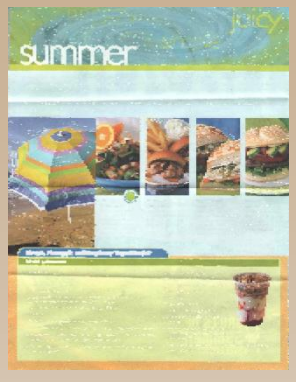

monophoto32

*Classification results for block sizes of 8x16 (Height x Width) pixels.

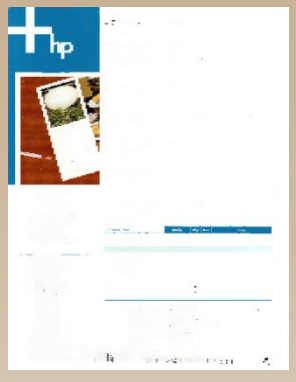
Sample Background Images



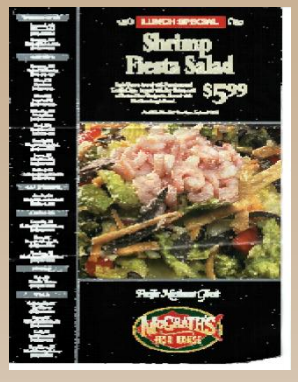
colormix300dpi



colormix110



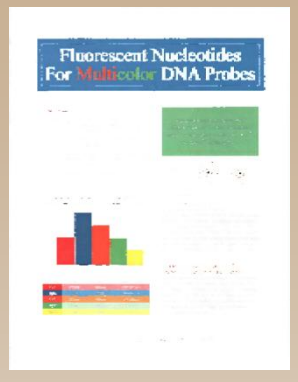
colormix114



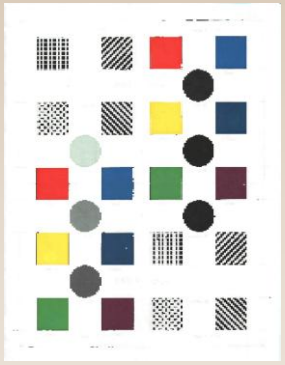
colormix117



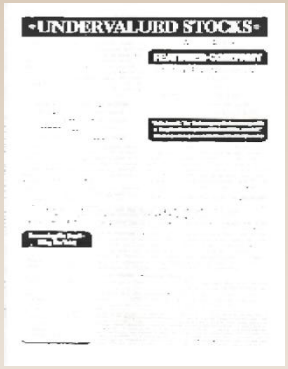
monomix9



colortext6



colortext16



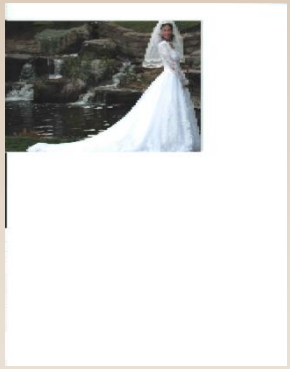
monotext12



colorpic11



monopic16



colorphoto6



monophoto32

- MCC compresses background images using JPEG Q25 at an average bit rate of 0.214 bpp (112:1 compression average)

Bit rate reduction achieved by MCC relative to that of JPEG



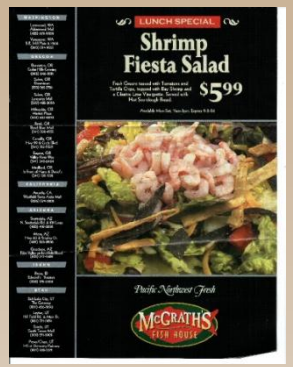
colormix300dpi
34%



colormix110
45%



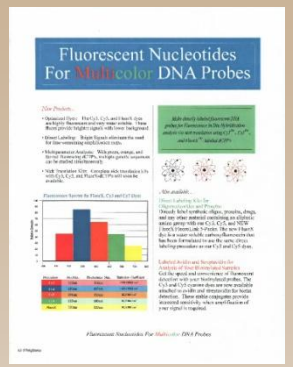
colormix114
55%



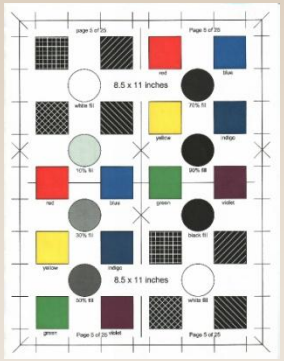
colormix117
41%



monomix9
39%



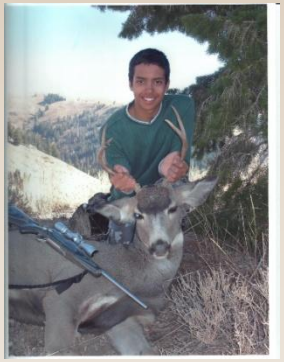
colortext6
47%



colortext16
43%



monotext12
60%



colorpic11
23%



monopic16
36%

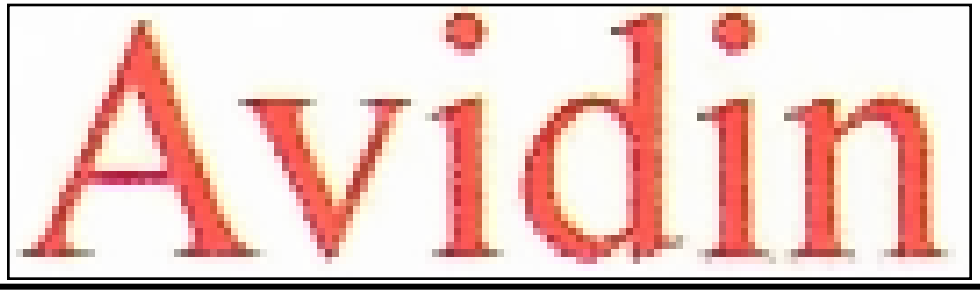
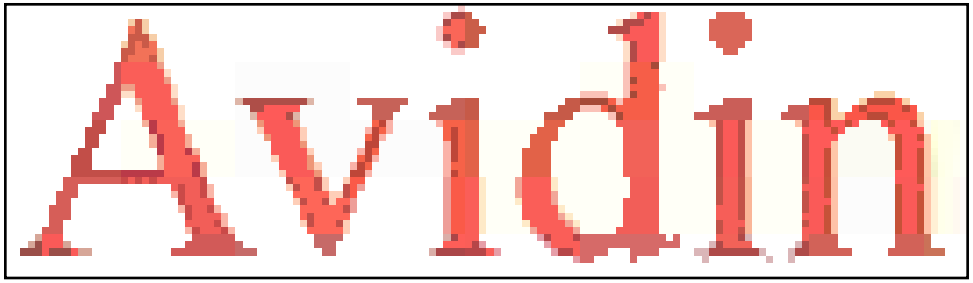


colorphoto6
10%

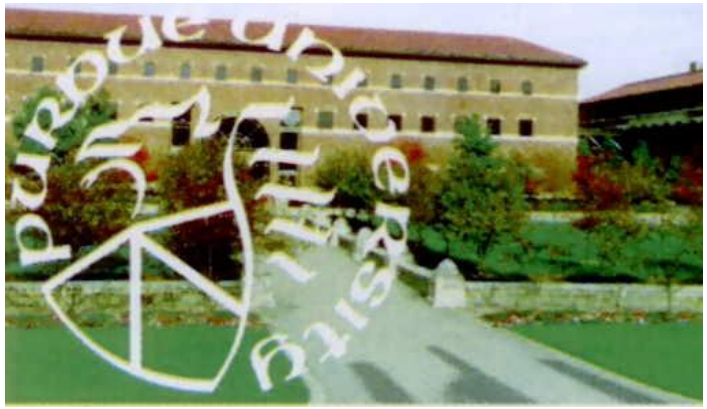
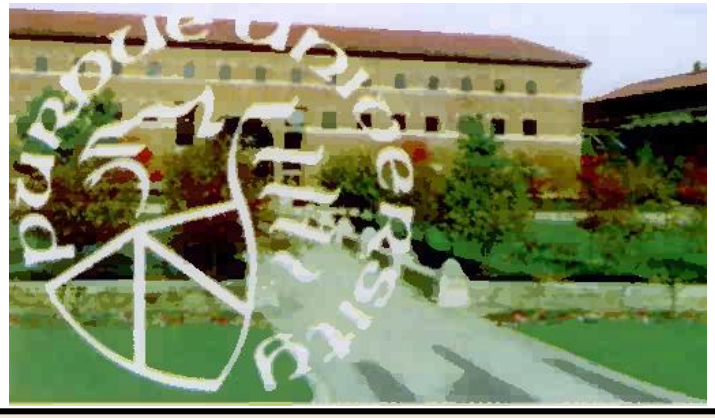


monophoto32
11%

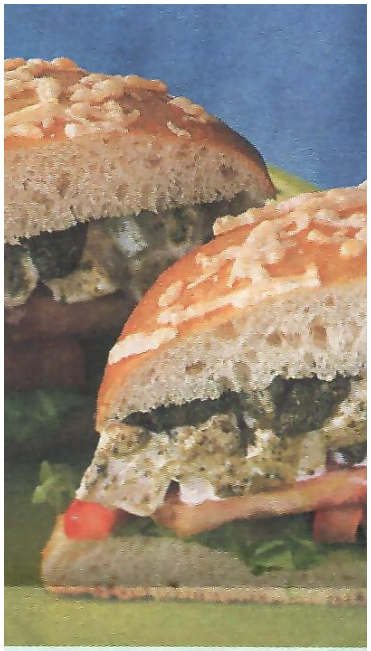
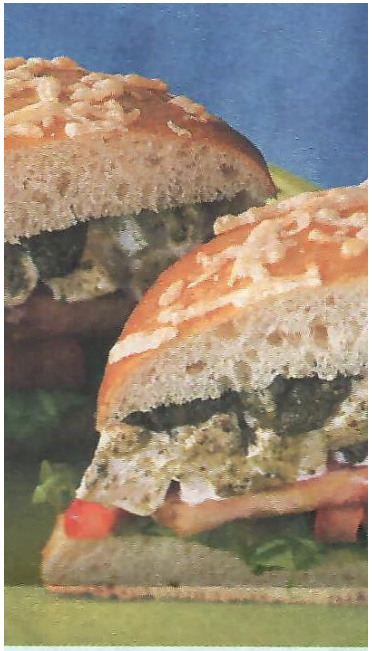
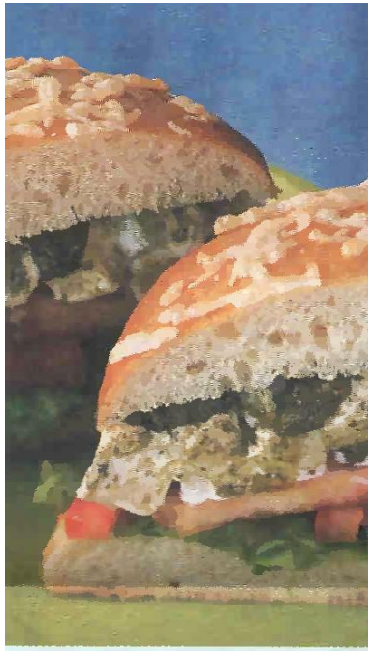
MCC and JPEG Comparison (color text)

Portion of Original Image		
JPEG	0.570 bpp (42:1 compression) 	
MCC	0.303 bpp (79:1 compression) 	

MCC and JPEG Comparison (color mixed)

Portion of Original Image		
JPEG		0.678 bpp (35:1 compression)
MCC		0.448 bpp (54:1 compression)

MCC and JPEG Comparison (color mixed)

Portion of Original Image	JPEG	MCC
	0.893 bpp (27:1)	0.491 bpp (49:1)
		
PESTO CHICKEN SALAD CIABATTA Chunks of chicken breast tossed with pesto mayonnaise along with sliced tomato and crisp romaine lettuce and an extra drizzle of pesto on our Parmesan topped ciabatta roll.	PESTO CHICKEN SALAD CIABATTA Chunks of chicken breast tossed with pesto mayonnaise along with sliced tomato and crisp romaine lettuce and an extra drizzle of pesto on our Parmesan topped ciabatta roll.	PESTO CHICKEN SALAD CIABATTA Chunks of chicken breast tossed with pesto mayonnaise along with sliced tomato and crisp romaine lettuce and an extra drizzle of pesto on our Parmesan topped ciabatta roll.

MCC and JPEG Comparison (mono text)

Portion of Original Image	**Answering the Need - Filling the Void!** **Molecular Pharmacology Limited (Nasdaq BB: MLPH)** has been to be an entirely new "anti-pain drug platform", and ultimately license and their timing couldn't be better! The fact that patents on nearly 100	
JPEG	**1.061 bpp (23:1 compression)** **Answering the Need - Filling the Void!** **Molecular Pharmacology Limited (Nasdaq BB: MLPH)** has been to be an entirely new "anti-pain drug platform", and ultimately license and their timing couldn't be better! The fact that patents on nearly 100	
MCC	**0.420 bpp (57:1 compression)** **Answering the Need - Filling the Void!** **Molecular Pharmacology Limited (Nasdaq BB: MLPH)** has been to be an entirely new "anti-pain drug platform", and ultimately license and their timing couldn't be better! The fact that patents on nearly 100	

MCC and JPEG Comparison (color mixed)

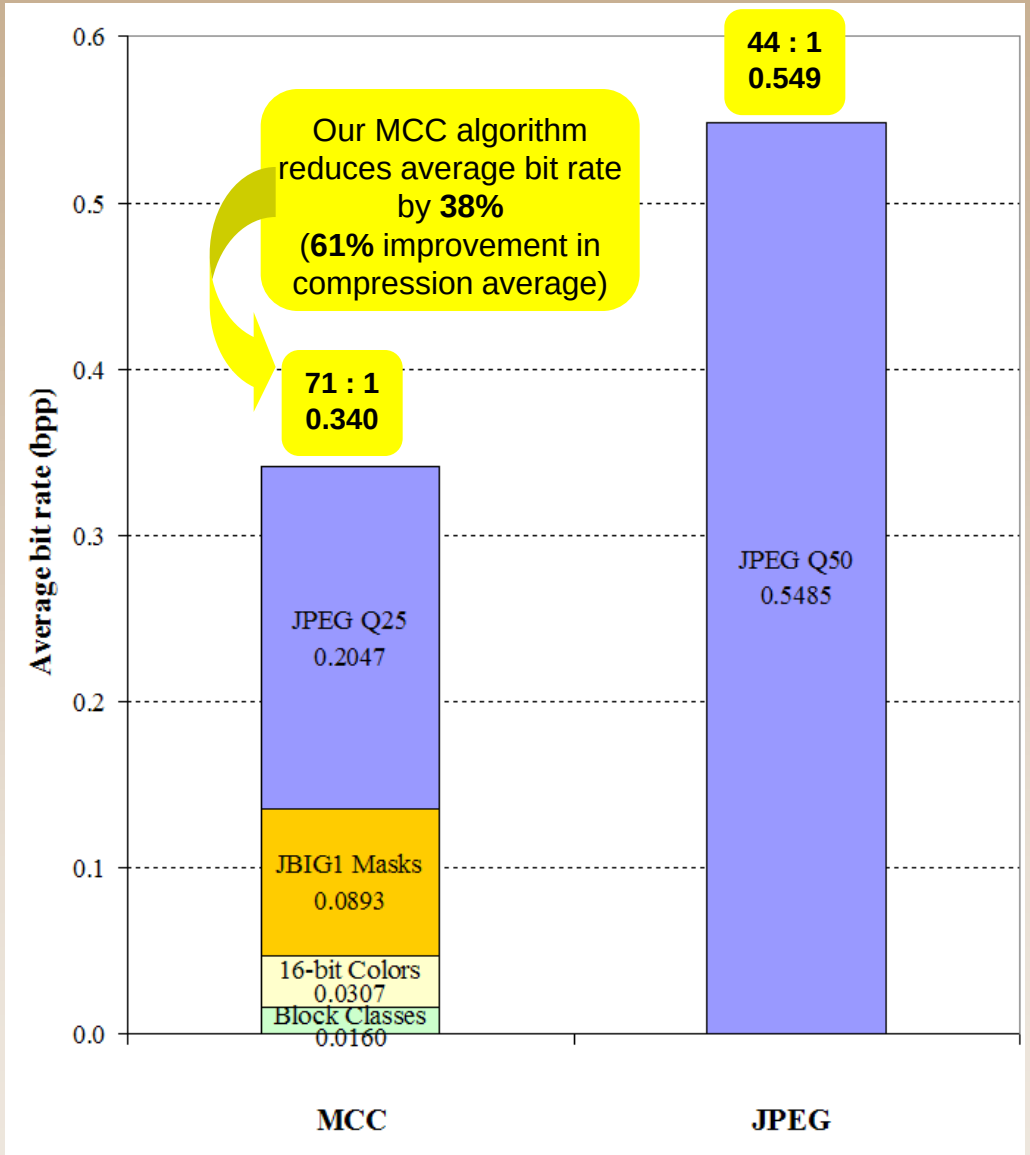
Portion of Original Image	
JPEG	0.678 bpp (35:1 compression) 
MCC	0.448 bpp (54:1 compression) 

MCC and JPEG Bit Rate Comparison for Each Image Category

Image Type	# of images	JPEG		MCC			
		Avg. bit rate (bpp)	Compression ratio average	Avg. bit rate (bpp)	Compression ratio average	Avg. bit rate reduction	Compression gain average
mono text	100	0.544	44:1	0.291	82:1	46%	87%
color text	100	0.529	45:1	0.307	78:1	42%	73%
mono mixed	100	0.683	35:1	0.383	63:1	44%	78%
color mixed	100	0.632	38:1	0.367	65:1	42%	72%
mono photo/picture	100	0.416	58:1	0.324	74:1	22%	28%
color photo/picture	100	0.487	49:1	0.370	65:1	24%	32%

Image Type	# of images	JPEG		MCC			
		Avg. bit rate (bpp)	Compression ratio average	Avg. bit rate (bpp)	Compression ratio average	Avg. bit rate reduction	Compression gain average
text	200	0.536	45:1	0.299	80:1	44%	80%
mixed	200	0.658	36:1	0.375	64:1	43%	75%
photo/picture	200	0.451	53:1	0.347	69;1	23%	30%

MCC and JPEG Bit Rate Comparison



Conclusion

- MCC can be easily implemented in imaging pipeline hardware
  - Uses only an 8 row buffer of pixels
  - Employs a simple minimal MSE-based block classification algorithm
- MCC sharpens text and reduces halo around text and vector objects
- MCC achieves a 38% average bit rate reduction relative to that of JPEG at similar quality, that is a 61% compression average gain
  - Relative bit rate reduction is higher for text (44%) and mixed images (43%)
- Summary compression efficiency:
  - MCC – 0.340 bpp (71:1 compression)
  - JPEG – 0.549 bpp (44:1 compression)

THANK YOU!

Questions?