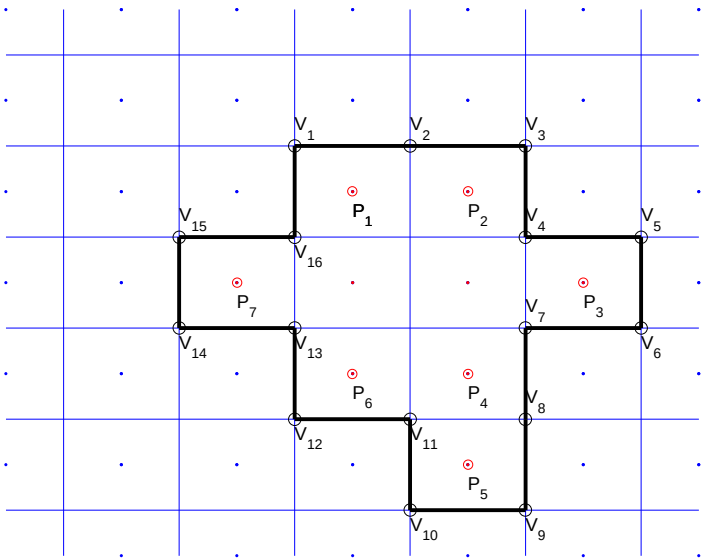
Optimal structure of memory models for lossless compression of binary image contours

Ioan Tabus, Septimia Sarbu

Department of Signal Processing
Tampere University of Technology
Finland

Contours of binary images

- Pixels coordinates on a square grid (subset of $\mathbb{Z} \times \mathbb{Z}$ lattice)
- In a binary image each pixel has a binary value (defining foreground and background)
- Contour: pixels from the foreground that are neighbors of the background pixels in 4-connectivity



- Image and Contour are in a one-to-one mapping

Contours of binary images

- Contour:
  - sequence $\mathcal{P} = [\underline{P}_1, \ldots, \underline{P}_N]$ of points $\underline{P}_t = (x_t, y_t)$
  - each pair of consecutive points $(\underline{P}_t, \underline{P}_{t+1})$ is connected in 8-connectivity.
  - $(x_{t+1}, y_{t+1}) = (x_t, y_t) + (i, j)$ with $i, j \in \{-1, 0, 1\}$ and $(i, j) \neq (0, 0)$.

μ	(i, j)	$(\lfloor \cos \mu_t \frac{\pi}{4} \rceil, \lfloor \sin \mu_t \frac{\pi}{4} \rceil)$
0	$(1, 0)$	$(\lfloor \cos 0 \rceil, \lfloor \sin 0 \rceil)$
1	$(1, 1)$	$(\lfloor \cos \frac{\pi}{4} \rceil, \lfloor \sin \frac{\pi}{4} \rceil)$
2	$(0, 1)$	$(\lfloor \cos \frac{2\pi}{4} \rceil, \lfloor \sin \frac{2\pi}{4} \rceil)$
3	$(-1, 1)$	$(\lfloor \cos \frac{3\pi}{4} \rceil, \lfloor \sin \frac{3\pi}{4} \rceil)$
4	$(-1, 0)$	$(\lfloor \cos \frac{4\pi}{4} \rceil, \lfloor \sin \frac{4\pi}{4} \rceil)$
5	$(-1, -1)$	$(\lfloor \cos \frac{5\pi}{4} \rceil, \lfloor \sin \frac{5\pi}{4} \rceil)$
6	$(0, -1)$	$(\lfloor \cos \frac{6\pi}{4} \rceil, \lfloor \sin \frac{6\pi}{4} \rceil)$
7	$(1, -1)$	$(\lfloor \cos \frac{7\pi}{4} \rceil, \lfloor \sin \frac{7\pi}{4} \rceil)$

Five chain codes as memory models

- **Freeman F8-chain code**:
  - the integer scalar $\mu \in \{0, 1, \ldots, 7\}$
  - the 8-connectivity displacement is $(i, j) = \underline{f}(\mu) \stackrel{def}{=} (\lfloor \cos \mu \frac{\pi}{4} \rceil, \lfloor \sin \mu \frac{\pi}{4} \rceil)$,
  - Unique value $\mu_t \in \{0, \ldots, 7\}$ such that $(\lfloor \cos \mu_t \frac{\pi}{4} \rceil, \lfloor \sin \mu_t \frac{\pi}{4} \rceil)^T = \underline{P}_{t+1} - \underline{P}_t$,
  - The reconstruction recursion $\underline{P}_{t+1} = \underline{P}_t + \underline{f}(\mu_t)$.
  - Equivalent representations: $\underline{P}_1, [\mu_1, \ldots, \mu_{N-1}] \Leftrightarrow [\underline{P}_1, \ldots, \underline{P}_N]$.

Five chain codes as memory models

- **Differential Freeman AF8-chain code**:
  - Denote $\beta_{t-1} = (\mu_t - \mu_{t-1}) \mathrm{mod}\ 8$ (non-negative modulo 8)
  - The reconstruction of $[\underline{P}_1, \ldots, \underline{P}_N]$ given β_t is

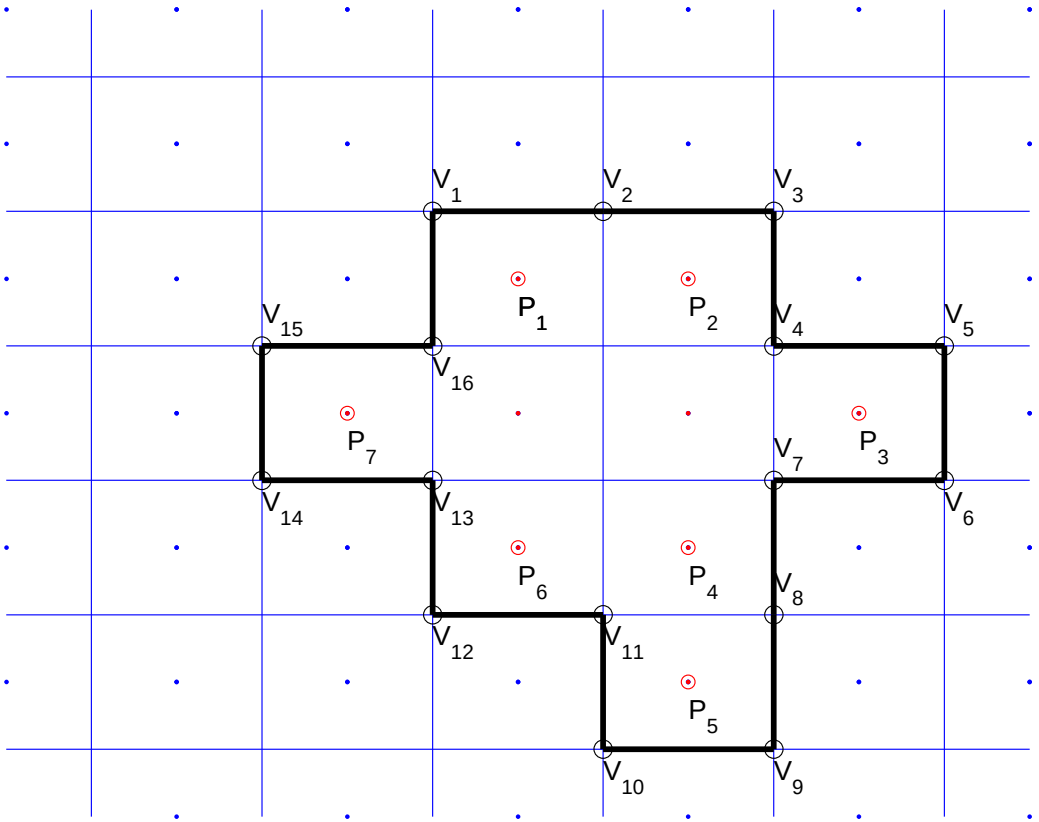
$$\underline{P}_{t+2} = \underline{P}_{t+1} + \lfloor B(\frac{\pi}{4}\beta_t)(\underline{P}_{t+1} - \underline{P}_t) \rceil$$

    where B is rotation matrix

$$B(\omega) = \begin{bmatrix} \cos(\omega) & -\sin(\omega) \\ \sin(\omega) & \cos(\omega) \end{bmatrix}$$

  - Equivalent representations:
    $\underline{P}_1, \underline{P}_2, [\beta_1, \ldots, \beta_{N-2}] \Leftrightarrow [\underline{P}_1, \ldots, \underline{P}_N]$.

Equivalent contour representations



Chain code
along pixels:
$P_1 P_2 \ldots P_7$

Chain code
along crack-vertices
$V_1 V_2 \ldots V_{16}$

Five chain codes as memory models

- **Freeman translated Ft4-chain code**: $\gamma \in \{0, \ldots, 3\}$
  - A first order model for $\underline{V}_1, \ldots, \underline{V}_M$:
    $\underline{V}_{t+1} = \underline{V}_t + \underline{g}(\gamma_t)$,
  - 4-connectivity displacements, $\underline{g}(\gamma) \stackrel{def}{=} (\cos\gamma\frac{\pi}{2}, \sin\gamma\frac{\pi}{2})$
  - Equivalent representations:
    $\underline{V}_1$, $[\gamma_1, \ldots, \gamma_{N-1}] \Leftrightarrow [\underline{V}_1, \ldots, \underline{V}_N]$.
- **Differential Freeman translated AFt4-chain code**:
  $\delta \in \{0, 1, 3\}$
  - Denote $\delta_{t-1} = (\gamma_t - \gamma_{t-1}) \bmod 4$
  - The reconstruction of $[\underline{V}_1, \ldots, \underline{V}_N]$ given δ_t is

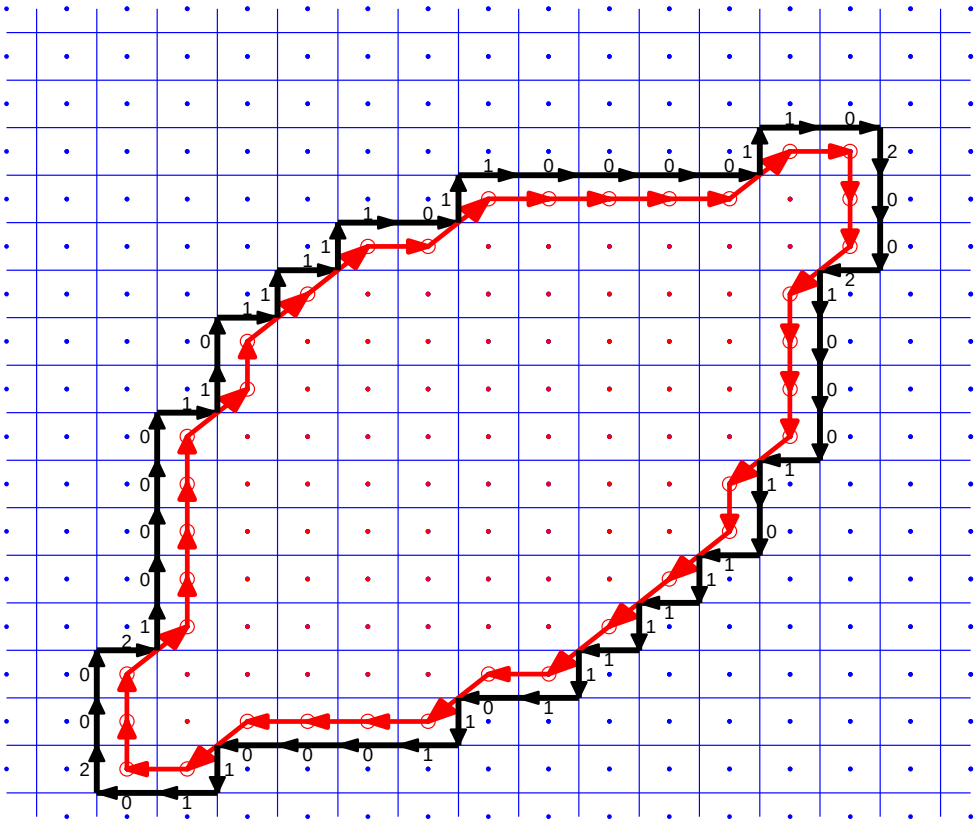
$$\underline{V}_{t+2} = \underline{V}_{t+1} + B(\frac{\pi}{2}\delta_t)(\underline{V}_{t+1} - \underline{V}_t)$$

  - Equivalent representations:
    $\underline{V}_1, \underline{V}_2$, $[\delta_1, \ldots, \delta_{N-2}] \Leftrightarrow [\underline{V}_1, \ldots, \underline{V}_N]$.

Five chain codes as memory models

- **3OT-chain code**: $\alpha \in \{0, 1, 2\}$
  - Similar to AFt4-chain code.
  - It has two state variables, storing the sense of movement along the horizontal and vertical direction (orthogonal movements).
  - Code 0 = no change in direction.
  - Code 1 = change of direction but keep the same old sense.
  - Code 2 = change of direction and change the old sense.
  - Virtually very long memory.
  - Equivalent representations:
    $\underline{V}_1, \underline{V}_2, [\alpha_1, \ldots, \alpha_{N-2}] \Leftrightarrow [\underline{V}_1, \ldots, \underline{V}_N]$.

Best contour representations



Chain code
along crack-vertices
$V_1 V_2 \ldots V_{54}$
Code 3OT =
= 10200210001101111110 ...

Chain code
along pixels:
$P_1 P_2 \ldots P_{37}$
Code AF8=
=120170001 ...

Applications of contour compression

- Good compression of binary images is equivalent to good compression of contours
- Compression of contours:
  - important in itself
  - provides statistical models of the objects. Interpretation as minimum description length (MDL) models
  - produces features for pattern recognition
- Representation of contours for image segmentation (color or graylevel) is well related but not identical

State of the art

- Many existing compression methods:
  - Fixed to variable (Huffman coding)
  - Variable to fix coding (define new symbols for alphabet extensions)
  - Variable to variable (runlength)
  - Context trees for AF8 - used for map (thin) contours

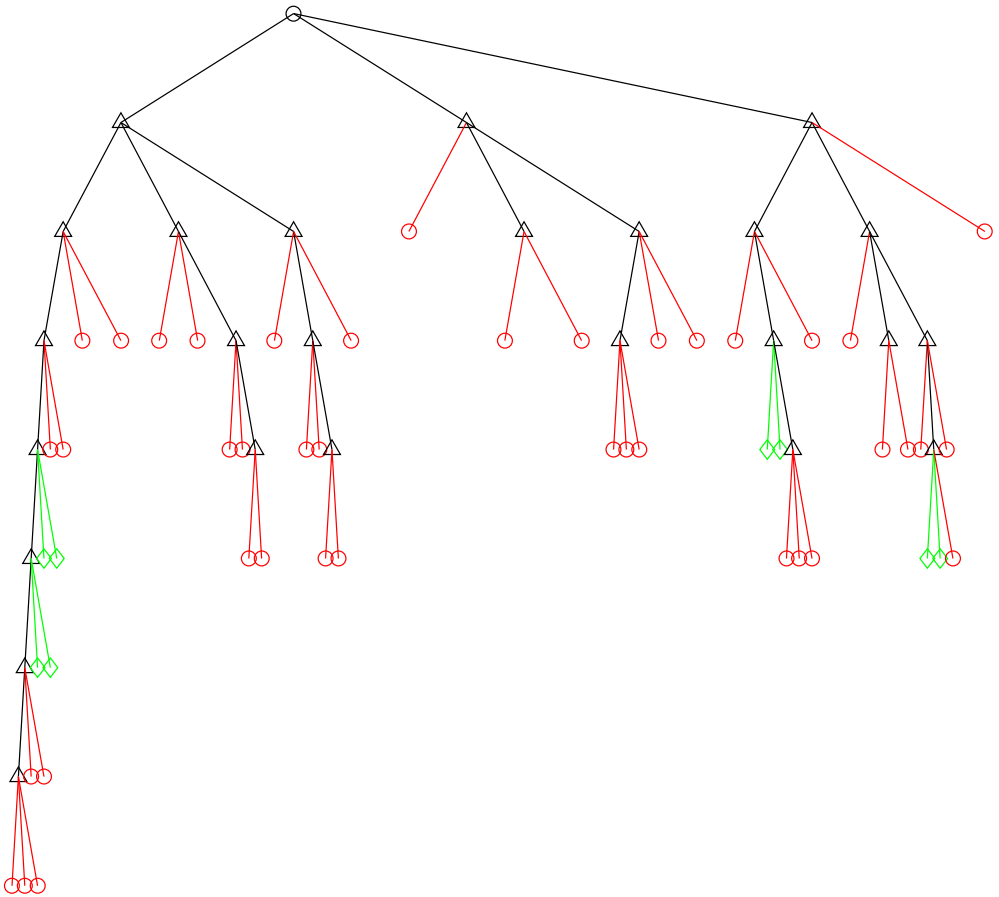
Contour modeling

- Questions:
  - What is the nature of a suitable image model for contour modeling?
  - Are $Prob(V_t|V_{t-1},\ldots,V_{t-k})$ and $Prob(P_t|P_{t-1},\ldots,P_{t-k})$ equivalent?
- Various model classes:
  - Geometrical models (pixel chain, vertex chain)
  - Time series (chain codes)
    - first order models
    - second order models
  - Higher order models:
    - Markov models of order k: $P(\alpha_t|\alpha_{t-1},\ldots,\alpha_{t-k})$,
    - Tree models having variable context length k_s: $P(\alpha_t|\alpha_{t-1},\ldots,\alpha_{t-k_s})$

Compression strategies

- Goal of modelling for compression: Define such models with which most of the predictive (coding) distributions used at $1, 2, \ldots, t, \ldots,$ are as skewed as possible.
- Context tree with poooling of contexts
- Compressor
  Contour $\Rightarrow$ Time series $\Rightarrow$ Coding distributions $\Rightarrow$ Arithmetic coding $\Rightarrow$ Compressd file
- Decompressor: Steps in reversed order

Context tree with poooling of contexts



The distributions at some leaves are merged. Equivalently, the distribution of the parent uses exclusion.

Optimization of the context tree structure

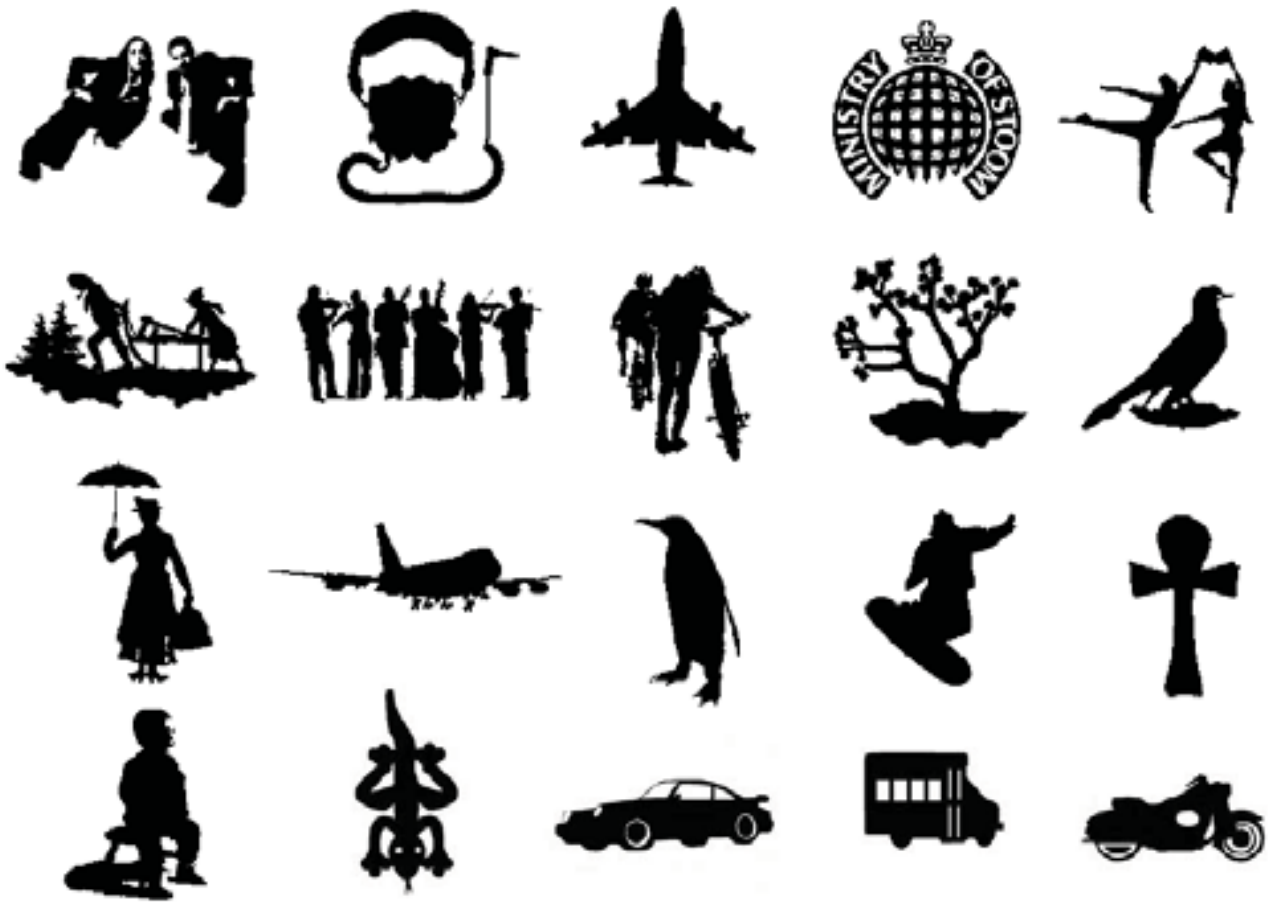
- Go through the whole sequence, $1 \leq t \leq T$, collect adaptively the codelengths $\mathcal{L}(Q, t)$ at each context $Q = (i_1, \ldots i_j)$
- With the final codelengths $\mathcal{L}(Q, T)$ do a bottom-up scanning of the tree
- Check the codelength at each parent, $\mathcal{L}_P = \mathcal{L}(Q, T)$, against the sum of codelengths $\mathcal{L}_C = \sum_{Q_i \text{ is child of } Q} \mathcal{L}(Q_i, T)$ at its children.
- If $\mathcal{L}_P > \mathcal{L}_C + n_s$ we decide to keep Q as an interior node and assign $\mathcal{L}_P \leftarrow \mathcal{L}_C + n_s$ to be used for all subsequent decisions of split of parents at lower depths.

Optimization of the context tree structure

- We collect at each node Q not only $\mathcal{L}(Q, t)$, but also codelengths corresponding to all possible grouping of the children of a parent node (2^{n_s} such grouping exists).
- When deciding a split we will have to compare parent codelength with the sum between the codelength at the pooled children and the rest of codelength of individual children. The structure penalization, (the term n_s from $\mathcal{L}_P > \mathcal{L}_C + n_s$) now becomes $n_s \log_2 3$.
- Label the nodes: 0 for nonterminal, 1 for independent terminal and 2 for pooled terminal. The stream of these ternary symbols is encoded by an adaptive Markov model of order 1.

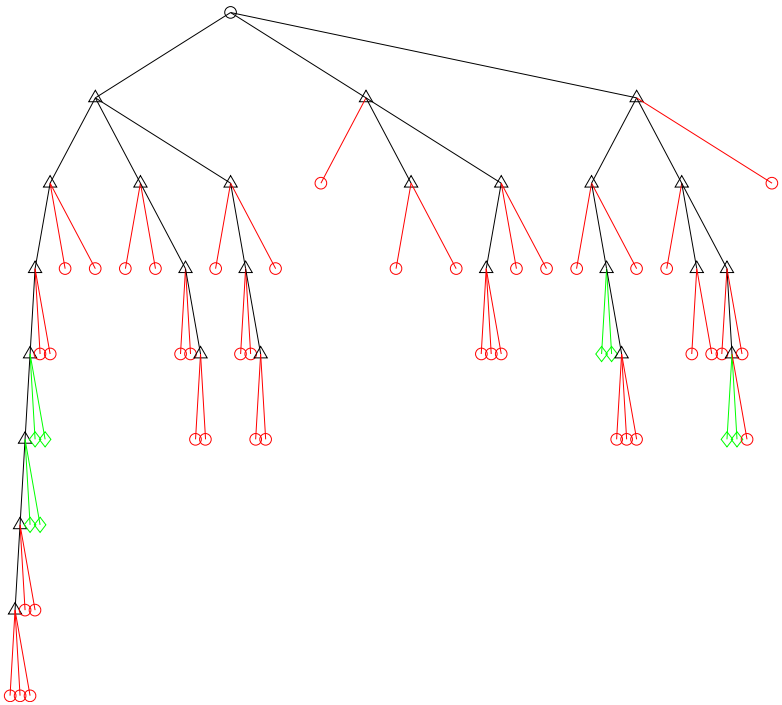
Sample Images

A data set of 100 binary images of various sizes (from 3Kb to 82Kb), many with interior holes, some with very nosiy contours

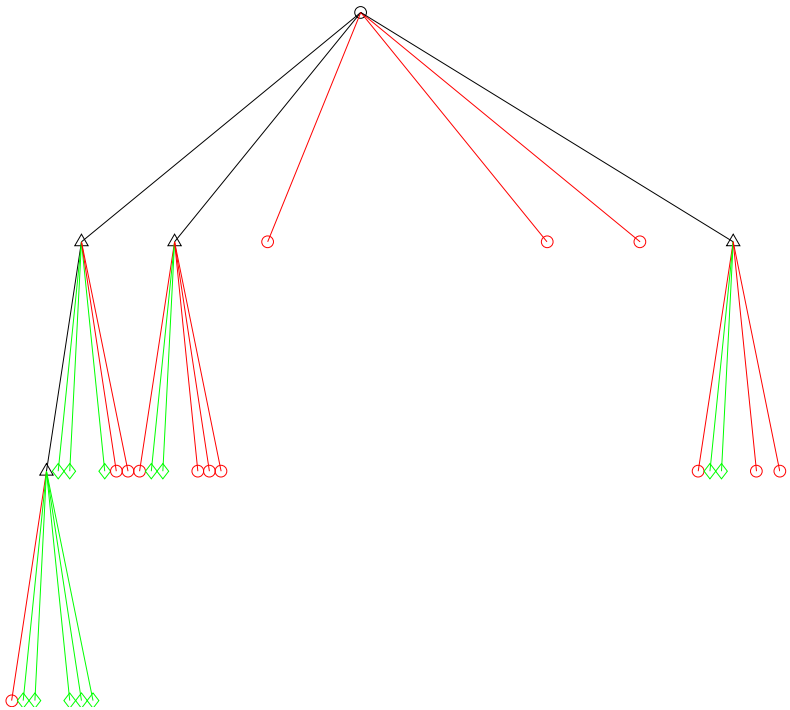


One example image with outer contour of $N = 5434$ pixels

Optimal context trees for example contour ($N = 5434$ pixels)

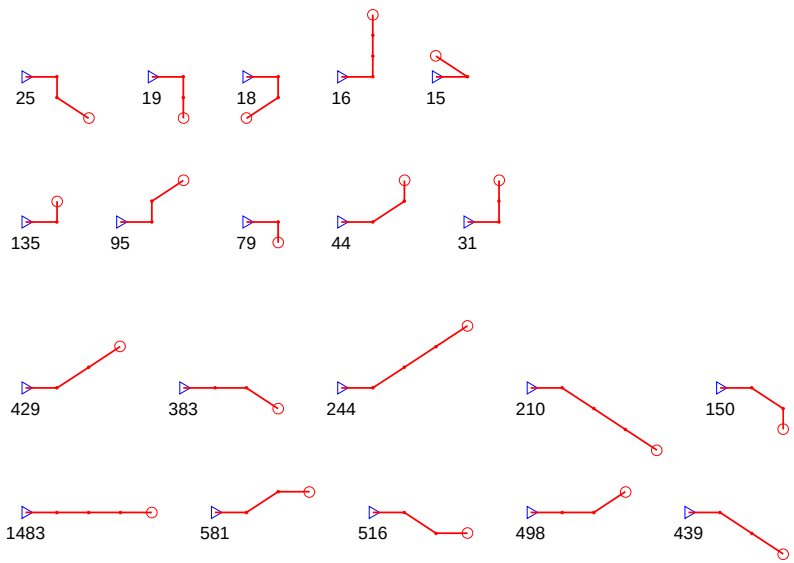


AFt4 chain code
$n_s = 3$ alphabet size
$M = 7214$ vertices
$n = 8$ maximum depth limit

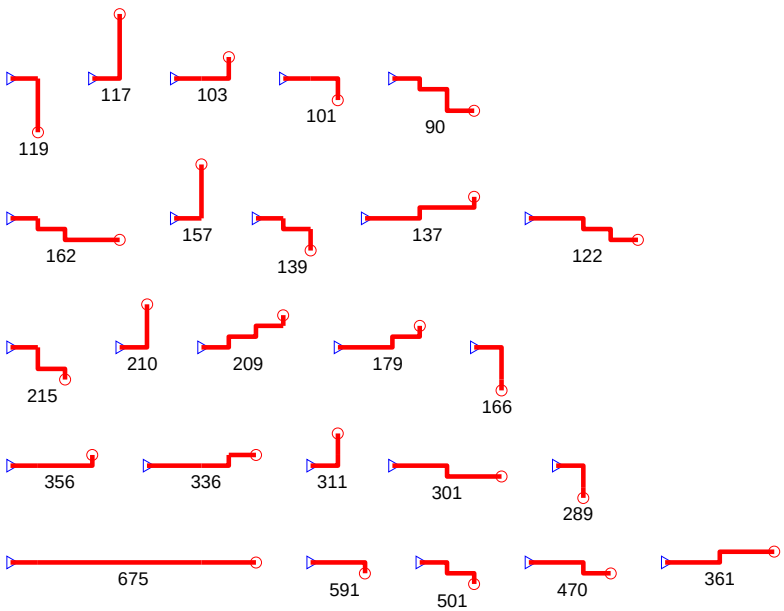


AF8 chain code
$n_s = 8$ alphabet size
$N = 5434$ pixels
$n = 3$ maximum depth limit

Optimal contexts for example contour ($N = 5434$ pixels)



AF8: The most frequently occurring twenty contexts $P_{t-1}, \ldots, P_{t-d_i}$.



AFt4: The most frequently occurring twenty contexts $V_{t-1}, \ldots, V_{t-d_j}$

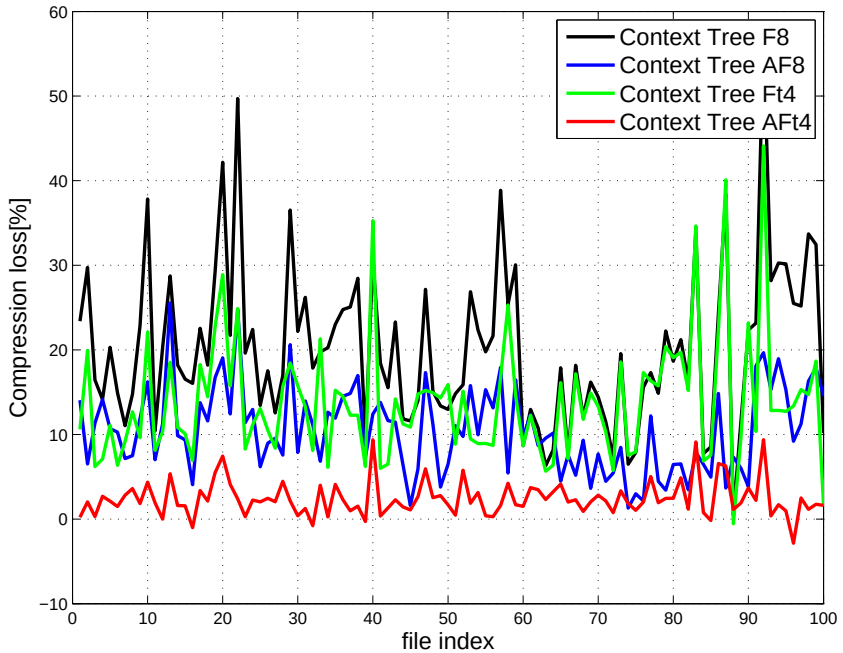
Optimal contexts for example contour ($N = 5434$ pixels)

Discussion:

- $P(\alpha_t|\alpha_{t-1}, \ldots, \alpha_{t-k})$ is not equivalent to $P(V_t|V_{t-1}, \ldots, V_{t-k})$.
- Some collapsing of contexts has been done: vertical contexts are collapsed with horizontal contexts, and even with diagonal contexts.
- This reduces the total number of contexts, but also may merge essentially different distributions.

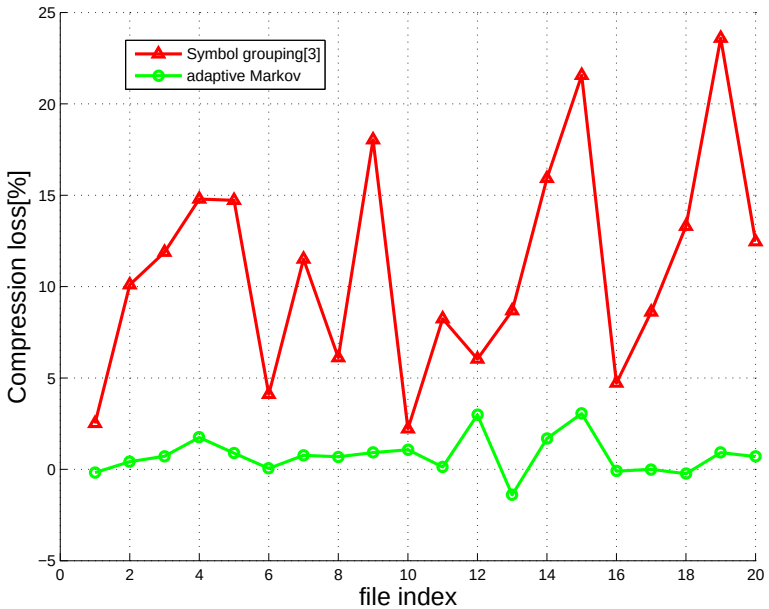
Context tree coding

Compression loss when using the chain codes F8,AF8,Ft4 and AFt4, instead of using 3OT



The ranking of the chain codes from best to worst is 3OT, AFt4, AF8, Ft4 and F8

Alternative compression methods: symbol grouping method and adaptive Markov coding



Compression loss when using the grouping method [2] and when using adaptive Markov distributions instead of using adaptive context tree encoding

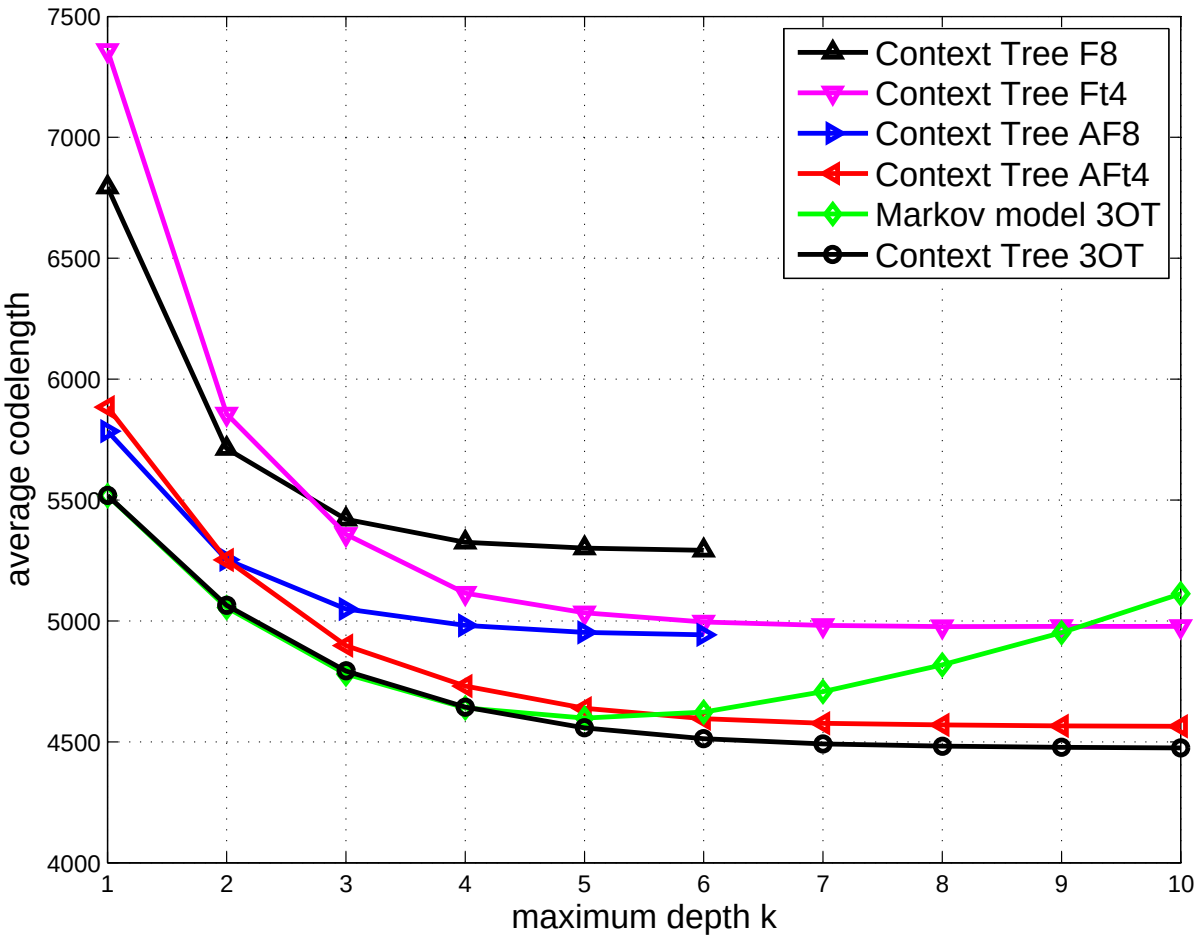
[2] S. Alcaraz-Corona, R.A. Neri-Calderon, and R.M. Rodriguez-Dagnino, "Efficient bilevel image compression by grouping symbols of chain coding techniques," *Optical Engineering*, vol. 48, no. 3, 2009.

Alternative compression methods for 3OT chain codes

image	BIC[1] Markov $\mathcal{L}(3)$	Adaptive Markov $\mathcal{L}(3)$	Adaptive Markov $\mathcal{L}(k^*)$	Adaptive Markov k^*	Context Tree $\mathcal{L}(k^*)$	Context Tree k^*
tools	4506	3908	3161	6	**3116**	10
logo	5133	4135	**3697**	5	3708	9
tree	**1285**	1310	1310	3	1324	3
portrait	5182	4999	4967	5	**4868**	7
car	3615	3289	**3066**	5	3068	8
Average (100 files)	5292	4787	4532		**4487**	

[1] S. Alcaraz-Corona and R.M. Rodriguez-Dagnino, "Bi-level image compression estimating the markov order of dependencies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 3, pp. 605–611, 2010.

Varying the maximum allowed depth



Average over 100 files

Conclusions

- Contour data can be represented in many equivalent ways.
- The statistical efficiency obviously favors the crack-vertex chains representations with tree context modelling.
- The structure of the optimal context tree model differs significantly from one file to another.
- Semi-adaptive methods should be prefered to static metods.
- The optimal contexts and their counts can provide potentially useful shape descriptors, being the minimum description length parameters of the best descriptions of the contours.