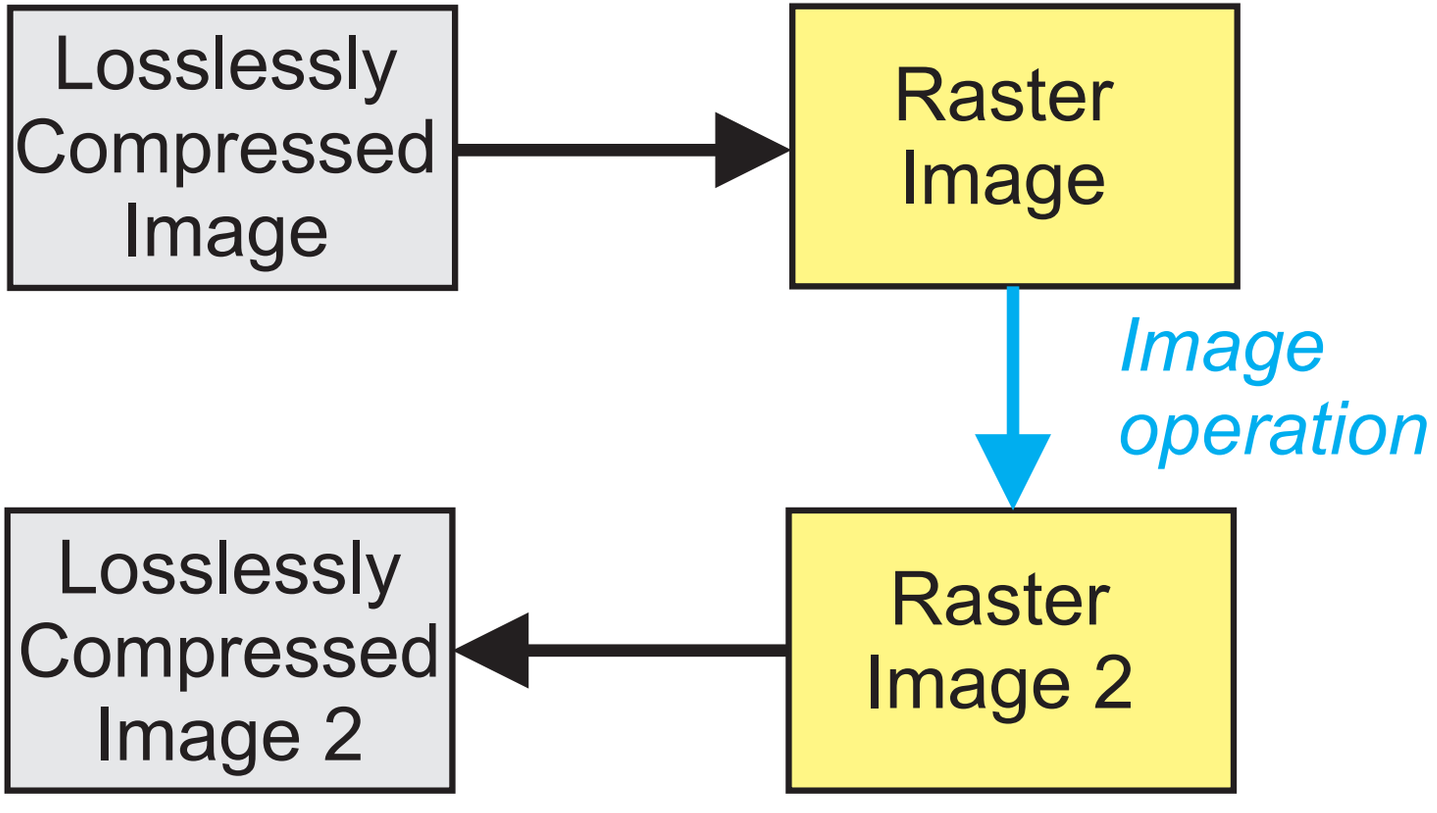
Corner Representation of Raster Images

Václav Hlaváč

Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Cybernetics
Center for Machine Perception
http://cmp.felk.cvut.cz/~hlavac, hlavac@fel.cvut.cz

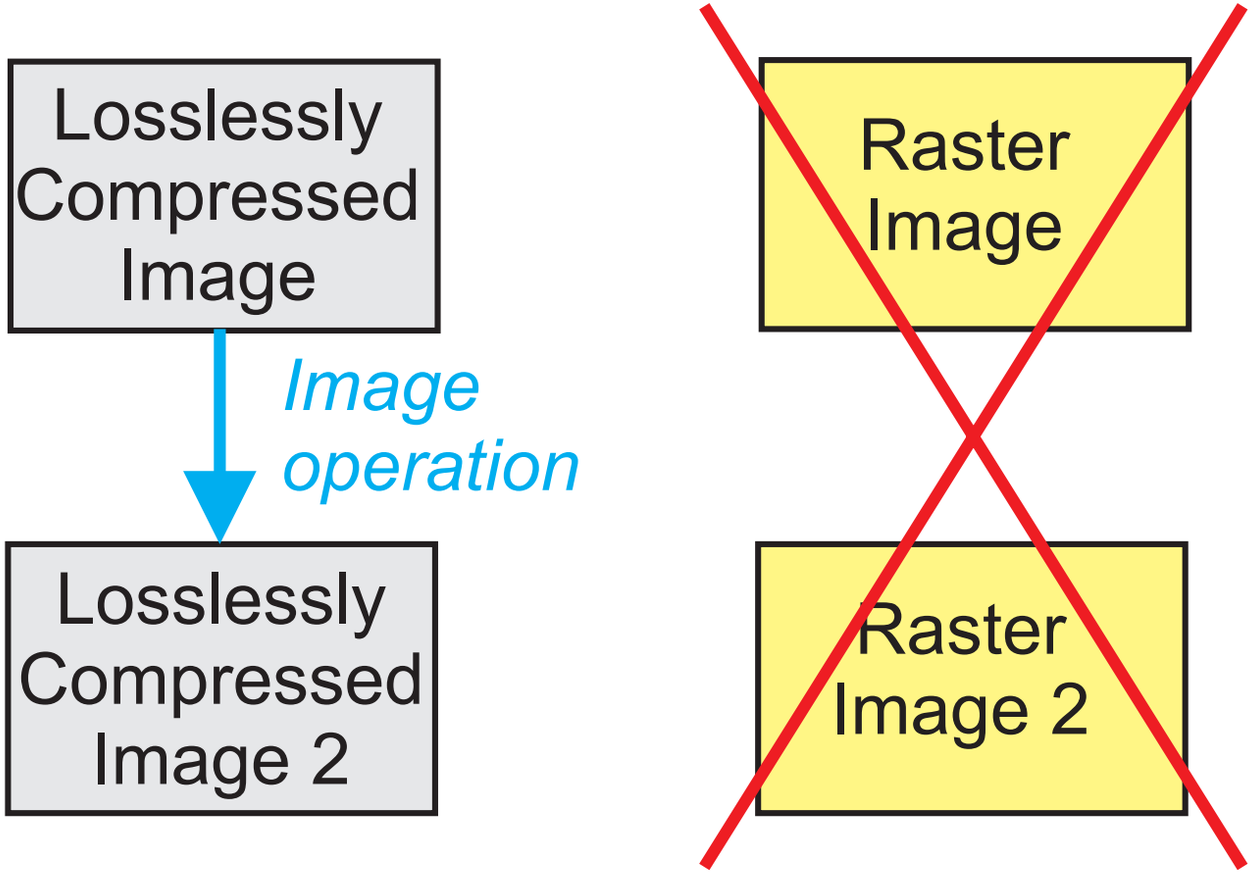
Decompression before applying operation

Standard way



Operations on compressed images directly

Desired way, avoid decompression



Corner representation, motivation

- Idea due to M.I. Schlesinger, Kiev, Ukraine, 1985.
- Remained almost unnoticed.
- I learned about it in 1996.
- There is a wide class of transformations that can be performed on corner compressed images.
- Let start with binary image $v(x, y) \rightarrow \{0, 1\}$.
  (1 stands for object, 0 otherwise).

Corner representation, example

Kanji character (meaning near from here), 360 $\times$ 345 pixels, 15,902 bytes. GZIPed 1,406 bytes.

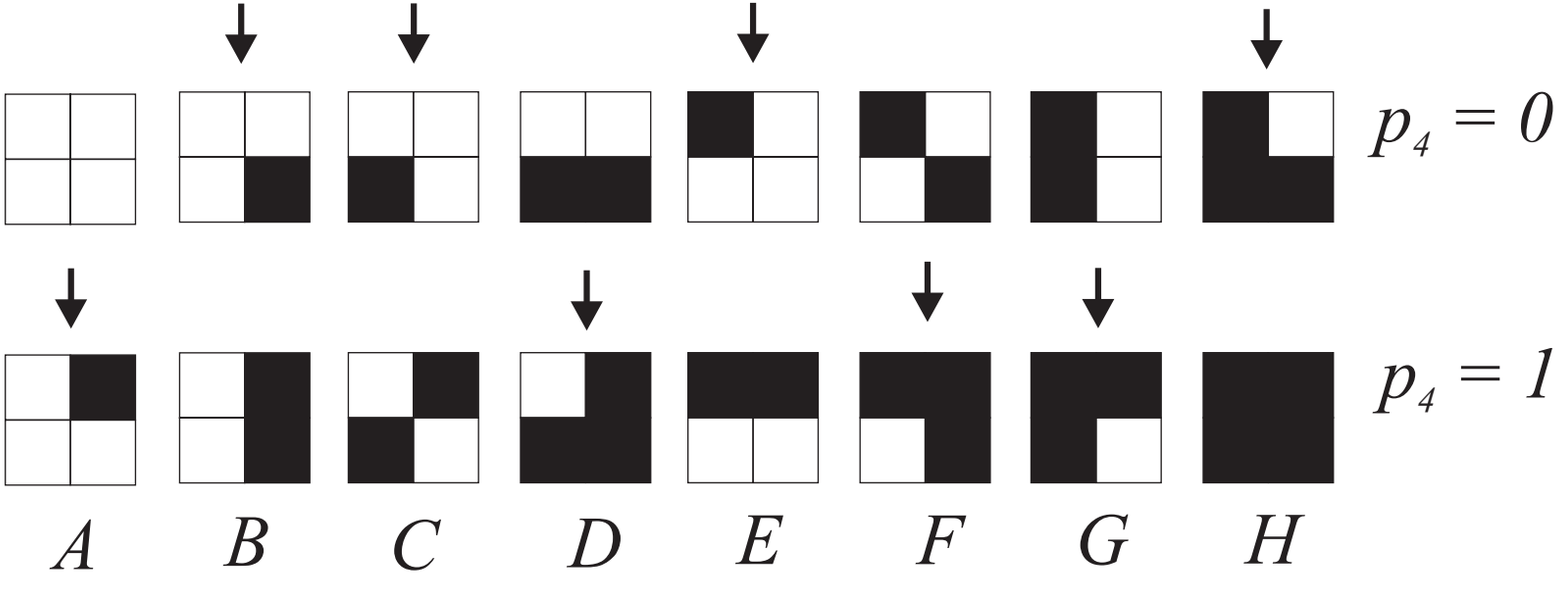
Corner compressed, 1,194 corners, 1,736 bytes, 10.9% of the original.

Overview

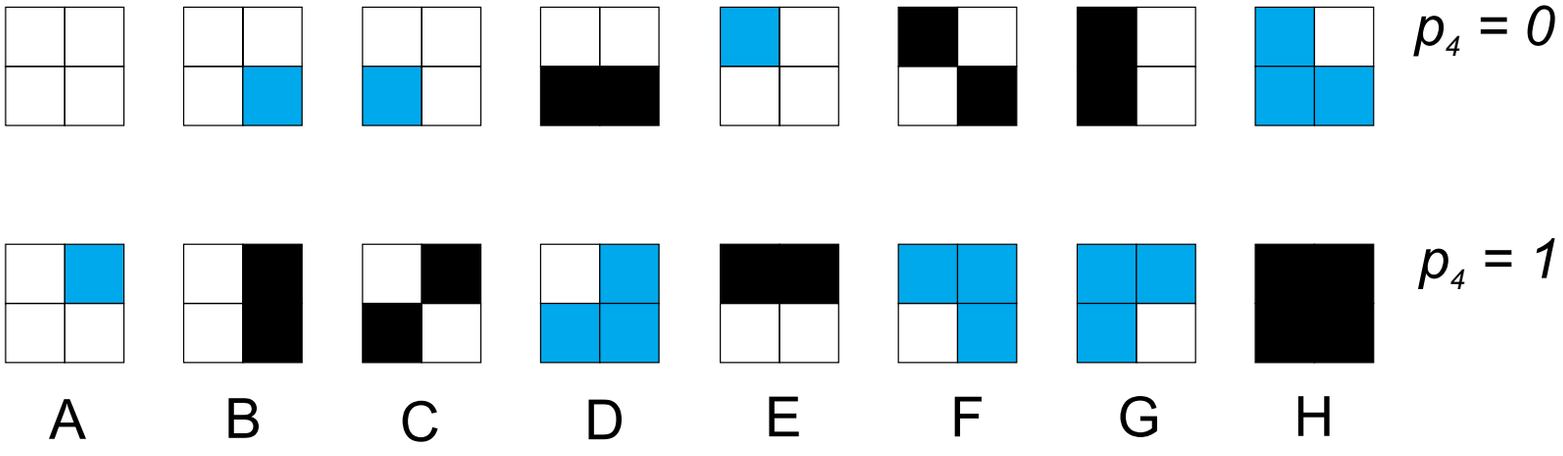
- A five student software engineering project with the stress to cooperative work in 1999.
- The goal was to create a public domain tool based on M.I. Schlesinger's corner representation of binary images.
- Implemented in C_{++}. It is a very rough implementation only.
- It should have been followed by a vectorization and a structural lines analysis project. This did not happen.
- A PhD thesis of Jaroslav Fojtík (2000] explored the idea to create an adaptive corner based predictor for image compression.
- The idea of generalization to n-dimensions came as the side effect.

Corner representation, idea 1

- Corner representation stores only residuals of the simple nonlinear predictor.
- Predictor probe $= 2 \times 2$ window only.
- Unsuccessful predictions correspond to corners.
- There are only $2^4 = 16$ possible configurations of the probe.



Corner representation, idea 2



- Corners K in the binary image v are given as
  $K(v) = \{(x, y) \mid k(x, y) = 1\}$,
  where: $k(x, y) = v(x, y) \oplus v(x - 1, y) \oplus v(x, y - 1) \oplus v(x - 1, y - 1)$.
- Odd number of black pixels in the probe indicates corner.

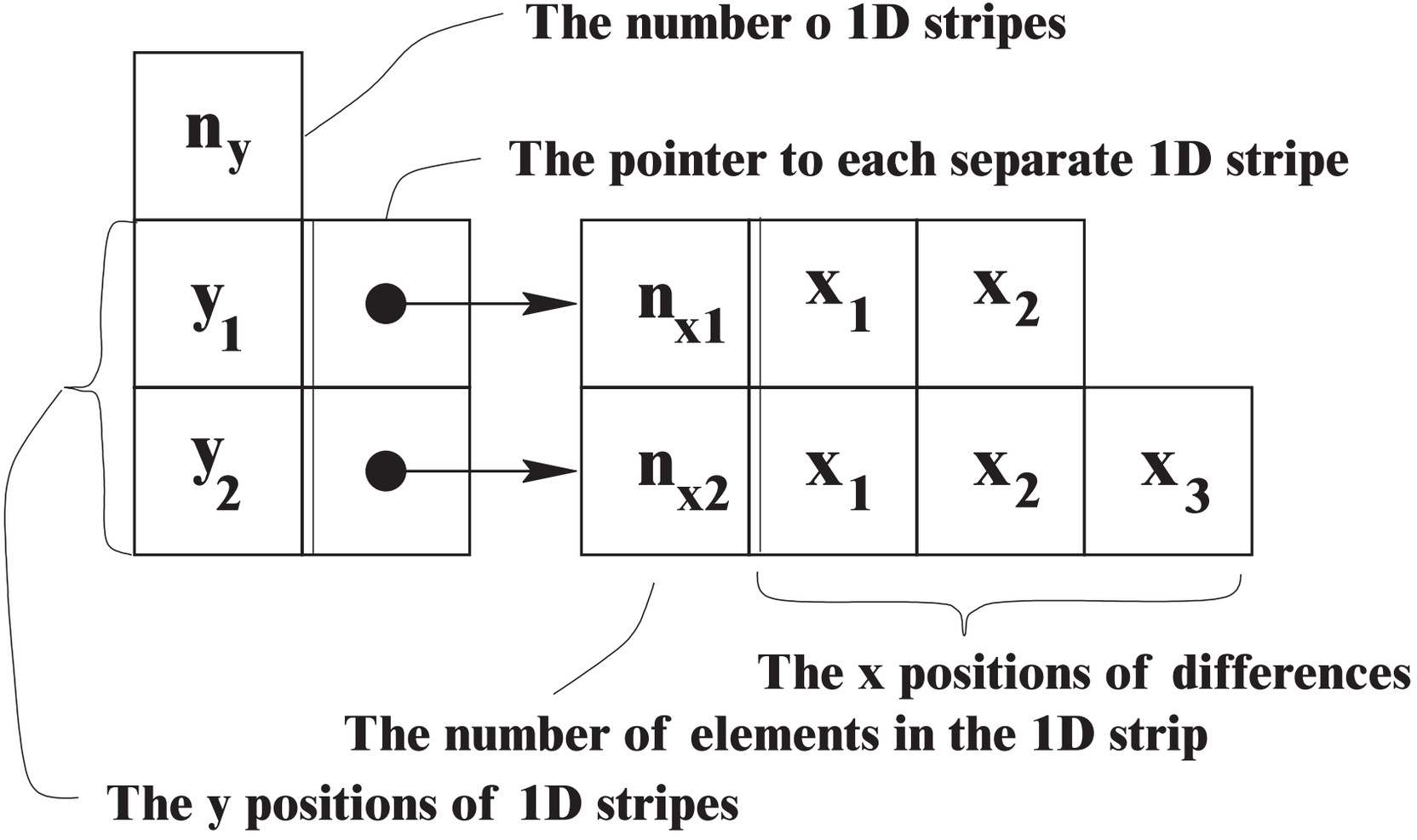
Compressing and decompressing

- The compression pass creates a corner representation. The binary image v is traversed by the 2×2 mask starting from the lower left position to the right and upward.
- The decompression pass starts again from the lower left corner of the compressed image and moves mask to the right and upward.
- Images in corner representation are not implemented as matrix of corners. The ordered list with pairs (x, y) corresponding to corners is used instead.

Compression into CR, an example

Decompression from CR, an example

How to store corners in a file?



Operations working in corner representation

General properties

- Operations (Scale, Shift, AND, OR) can be performed in $\mathcal{O}(n)$ time, where n is the sum number of corners in the image.
- XOR$(A, B) \equiv A \oplus B$ finds exclusive corners only. Complexity is $\mathcal{O}(n)$, where n is smaller from number of corners in images A, B.
- NOT can be performed in constant time.
- There is usually less corners than object pixels in the image. The operations are thus faster.

Shift, scale

Let $c(A)$ be the corners of an image A.

$$c(A) = c_1, \ldots, c_n = (x_1, y_1), \ldots, (x_n, y_n)$$

Shift $A' = A + (a, b), \quad a, b \in \mathcal{Z}$

for $i = 1$ to N **do**

$(x_i, y_i)' = (x_i, y_i) + (a, b)$

end for

Scale $A' = A \cdot s, \quad s \in \mathcal{R}$

for $i = 1$ to N **do**

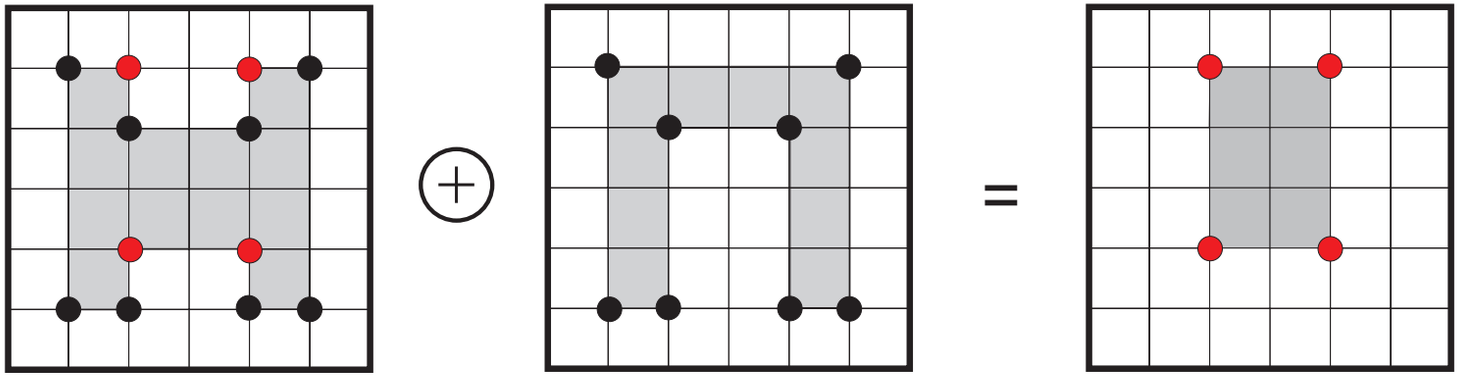
$(x_i, y_i)' = \text{round}((x_i, y_i) \cdot s)$

end for

Exclusive OR, $E = A \oplus B$

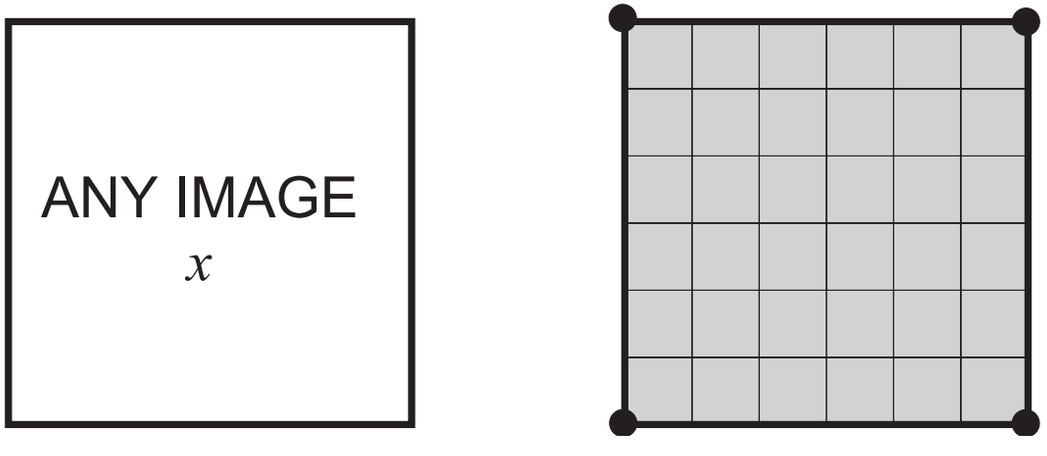
Let $\#_c(A)$ denote the number of corners in an image A.

- Algorithm $c(E) = c(A) \oplus c(B)$.
- The following holds $\#_c(E) \leq \#_c(A) + \#_c(B)$.
- The complexity of $\oplus$ depends on number of corners in images with less number of corners.



Negation, $\overline{A} = A$

- $\overline{A} = A \oplus 1$,
  where the number 1 represents the entirely black image.
- Four operations are needed only, i.e. the number of operations does not depend neither on the number of pixels nor number of corners.



Connected components counting 1

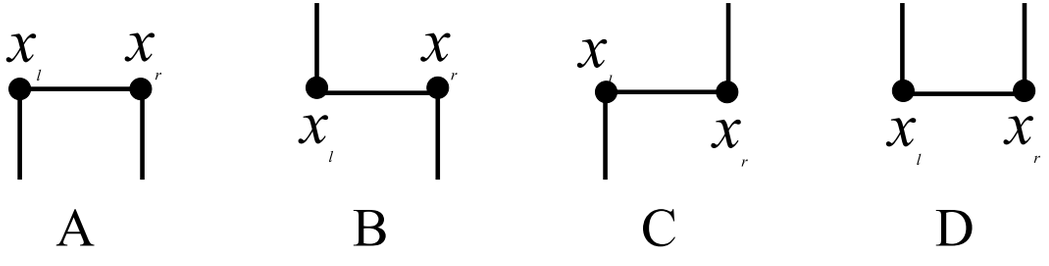
- Performed directly in corner representation. One pass suffices.
- Corners represent well the contour of the connected component (region).
- Principle of line-wise calculation:

  1. Finding contours;
  2. Determining which contour is inner and which outer;
  3. Update counter of connected components.

- Additional region features can be calculated (area, perimeter, moments, . . .).

Connected components counting 2

- Line-wise, pairs of corners are read from the left-hand side.
- 4 possible configurations possible only.



- It is decided if contour is inner one or outer one (recall Jordan curve, odd-even number of intersections).
- The connected component counter is updated.

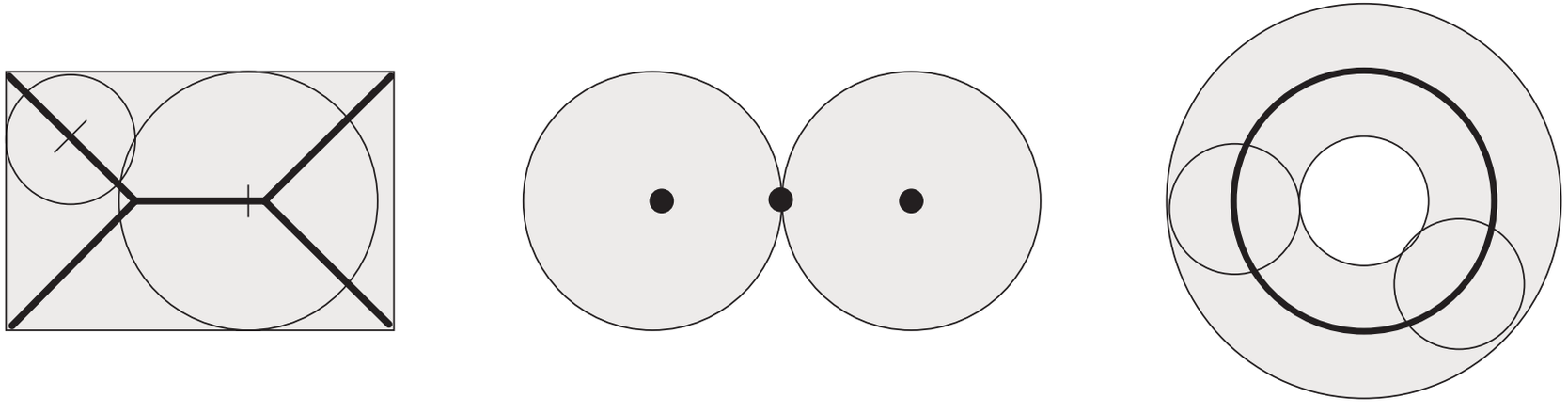
Coloring connected components (segmentation)

- 2 passes through corner represented image.
- Similar to counting connected components. The color attribute is added to each connected component in the first pass.
- Conflicts of colors are solved in the second pass.
- Use of coloring:
  - Selection of individual regions.
  - Filtering out regions of certain properties, typically small ones that correspond to noise.

Skeleton, the concept

Introduced by Blum (1964) by a grass fire scenario.

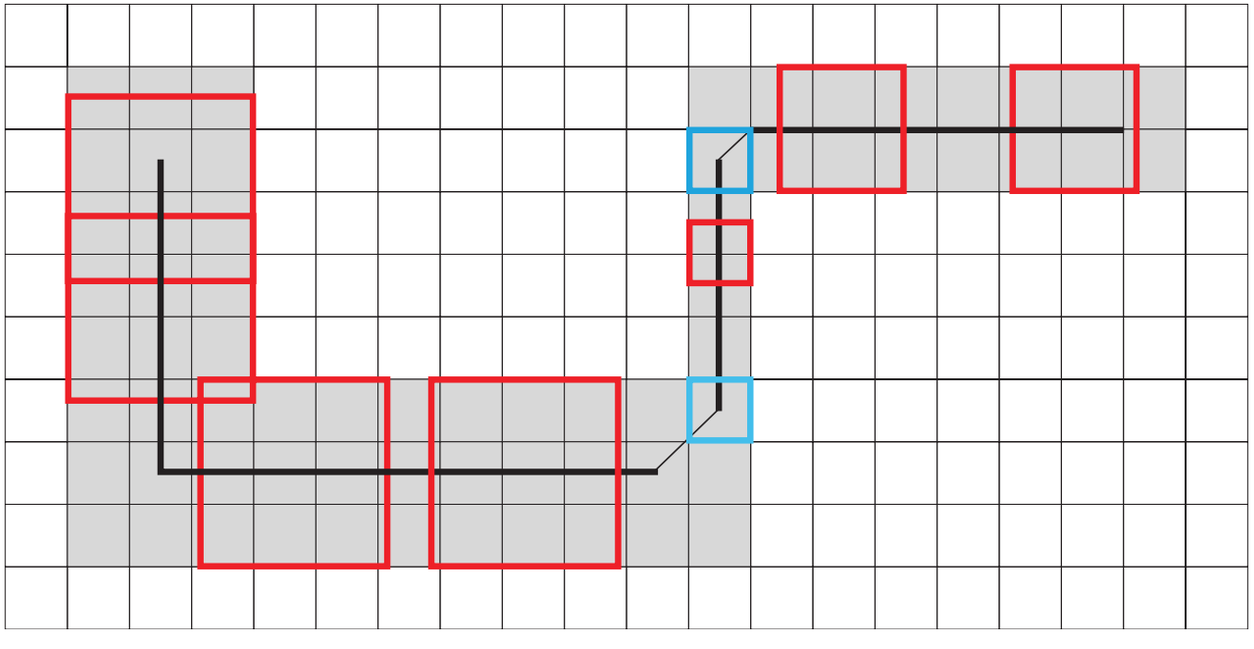
A skeleton of a binary object is traditionally defined as a union of centers of inscribed circles.



Two approaches: (a) Sequentional thinning; (b) Middle of the abscissa to the "opposite point of the contour".

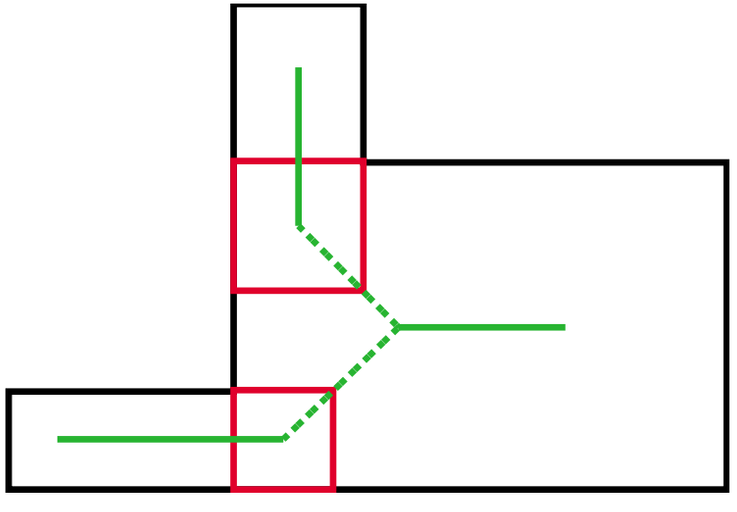
Skeletonization in corner represented image

- We define skeleton as union of centers of maximal squares.
- The basic skeleton consists of vertical and horizontal lines only. It is typically discontinuous.
- Auxiliary skeleton (direction modulo 45^o) join basic skeleton components to preserve topology.



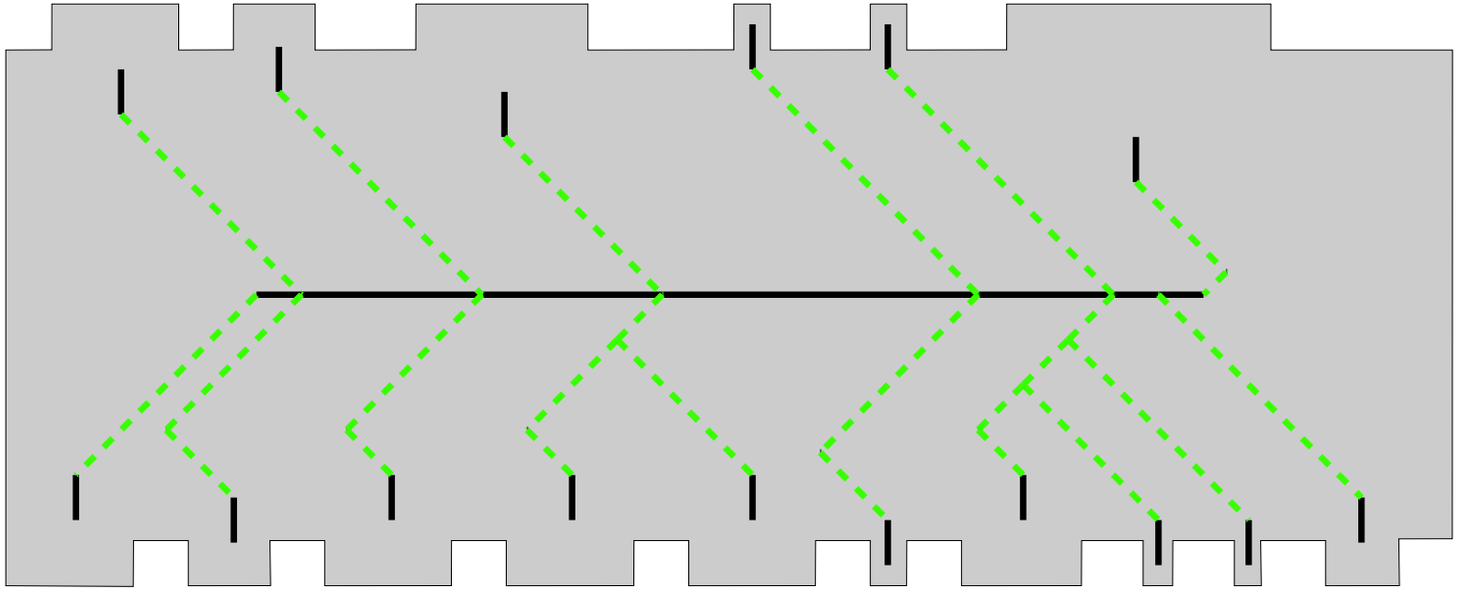
Calculation of the skeleton, “divide and conquer”

- The skeleton of a rectangle is trivial. The idea is to decompose the region into rectangles.
- Key to decomposition is the dissective square grows from concave (L-shape) corner and it is one of maximal squares.
- If such a maximal square does not exist then it is one of the closure of set of maximal squares.
- Dissective squares define auxiliary skeleton.



Ragged skeleton

- Real skeletons are ragged due to noise.
- The homomotopy of the region can be represented by a graph.
- Postprocessing of the graph allows to smooth (interpolate) the skeleton.



Our research contribution

- Use of idea of corner representation (adaptive predictor) to compress grey level images. Treated as stacked bit-planes.
- Generalization into three dimensions.