FIT VUT Brno

Digital Images and Formal Languages – 1st part

TID – course

Jiří Krajíček

12.11.2007

Digital Images and Formal Languages – 1st part

- Outline

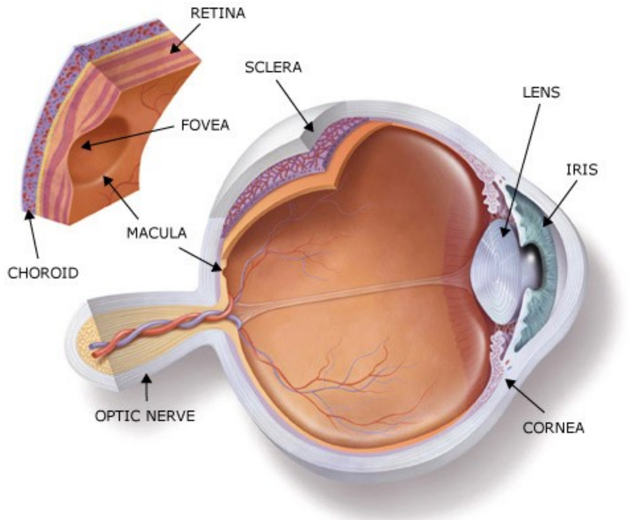
I. Motivation – our aim
II. Digital image – basic info
III. Different types of image formats
IV. Image representation by formal languages.
V. Black and white images.
VI. Fractal generation.
VII. Conclusion

Digital Images and Formal Languages – 1st part

- *Motivation:*
  - Human eye can recognize approximately 16.7 millions of colors.
  - Hence we do not usually require to store more information.
  - ***Digital image***
    - Information about image points represented in binary format.
    - We need to know:
      - How to represent image model in efficient way.
      - How to do operations like zoom, compression, filtering etc.

Digital Images and Formal Languages – 1st part

- *Image representation – main approaches:*
  - ***Raster graphics*** *(bitmap): divide image by raster (regular net) into elementary points (pixels), where each pixel at its position contains information about* the brightness of a given color .
    - *Our eye catch image like raster graphics (by retina).*
  - ***Vector graphics****: describe image by geometry primitives.*
    - *Our brain process image like vector graphics.*



Digital Images and Formal Languages – 1st part

- *Image representation – raster formats:*

How to represent any raster image?

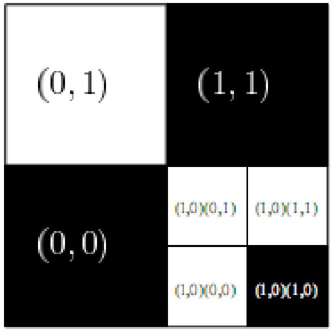
- We need to cover all color variants (16,7 million of colors) .
- Lets have RGB color model and 8 bits for each color (Red, Green, Blue).
- Then we obtain $\mathbf{2^8}$ (256) colors for each RGB color component:
  - 256 variants for Red,
  - 256 variants for Green
  - 256 variants for Blue.
- As result we obtain 256 * 256 * 256 → **16, 7** million of colors (its variants), called **True Color image**. This representation requires 24 bits for each pixel.

Digital Images and Formal Languages – 1st part

- *Image representation – raster formats:*
- **BMP** – directly stores pixel values according to RGB model. Most common is 24-bit BMP variant (as presented above).
- **GIF** – *Graphics Interchange Format, based on RGB,* uses lossless data compression based on patented LZW84 algorithm. May contain simple animation.
- **PNG** – *Portable Network Graphics,* based on RGBA (plus Alpha channel), uses lossless data compression based on patented LZW84 algorithm.
- **JPEG** – *Joint Photographic Experts Group*, JPEG is distinct from MPEG, uses data loss compression:

Digital Images and Formal Languages – 1st part

- *Image representation by formal languages:*
  - Since points in 2-dimensional space are represented by two coordinates (numbers), they can be represented by strings over an n^2–letter alphabet.
  - Sets of points (pixels) can be interpreted as image.
  - Regular sets can be used to represent geometrical objects and fractals.
  - First lets focus on bi-level images (black and white).

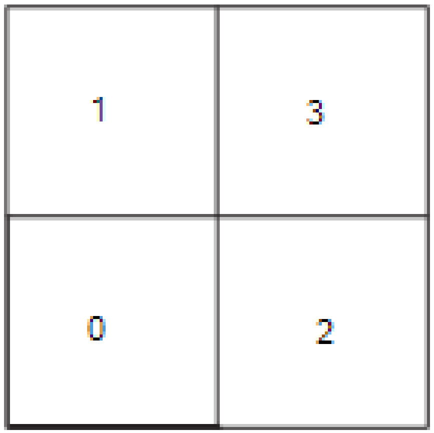


Digital Images and Formal Languages – 1st part

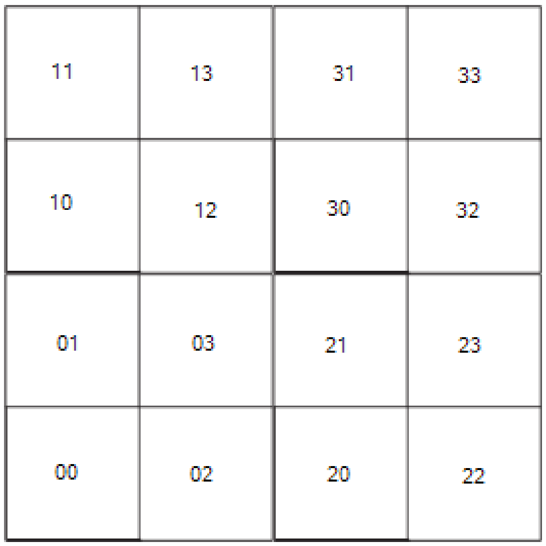
- *Image representation by formal languages:*
- Lets consider square images of resolution 2^n x 2^n .
  - Then we will need an alphabet Σ = {0, 1, 2, 3}.
  - Each pixel (point) is addressed by string of length ***n*** over Σ.
  - A pixel (sub-square) at resolution 2^n x 2^n corresponds to size 2^{-n} of the whole unit square.
  - ε – represents whole unit.

Digital Images and Formal Languages – 1st part

- *<u>Image representation</u> by <u>formal languages</u>:*
- *Example 1)*
  For *n* = 1. Here we obtain 4 pixel image:



- *Example 2)*
  For *n* = 2. Here we obtain 16 pixel image:



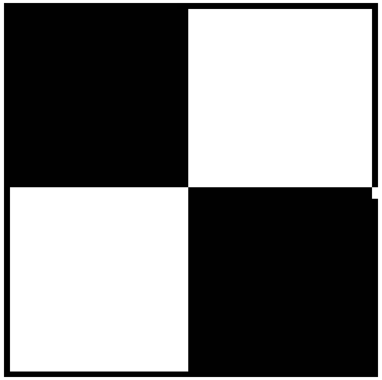
Digital Images and Formal Languages – 1st part

- *Image representation* by *formal languages*:
- Black and white images.
  - In order to specify a black and white image of resolution 2^m x 2^m, we need to specify Boolean function $\Sigma^m \to \{0, 1\}$.
  - Or alternatively we can just specify the set of black pixels, i.e. a language: $L \subseteq \Sigma^m$. This is called the ***image specification***.
  - Generally, multi-resolution black and white image is specified by language $L \subseteq \Sigma^*$, where $\Sigma = \{0, 1, 2, 3\}$, i.e. by addresses of black pixels.

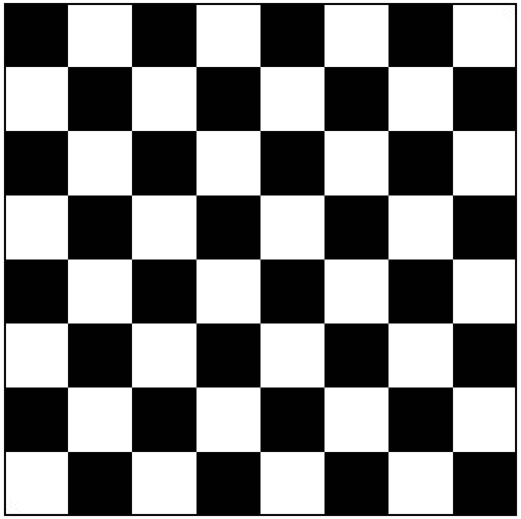
Digital Images and Formal Languages – 1st part

- *Image representation* by *formal languages*:

- *Example 3*)
  Lets have resolution 2^m x 2^m , where
  $m \geq 1$ and specification is defined by
  regular set: **{1, 2} $\boldsymbol{\Sigma}^{m-1}$** .
  Note: looks the same for all **resolutions**.



- *Example 4*)
  According to example 3, generally
  multi-resolution image of chess board (8 x 8) is
  described by regular set: **$\boldsymbol{\Sigma}^2$ {1, 2} $\boldsymbol{\Sigma}^*$**.



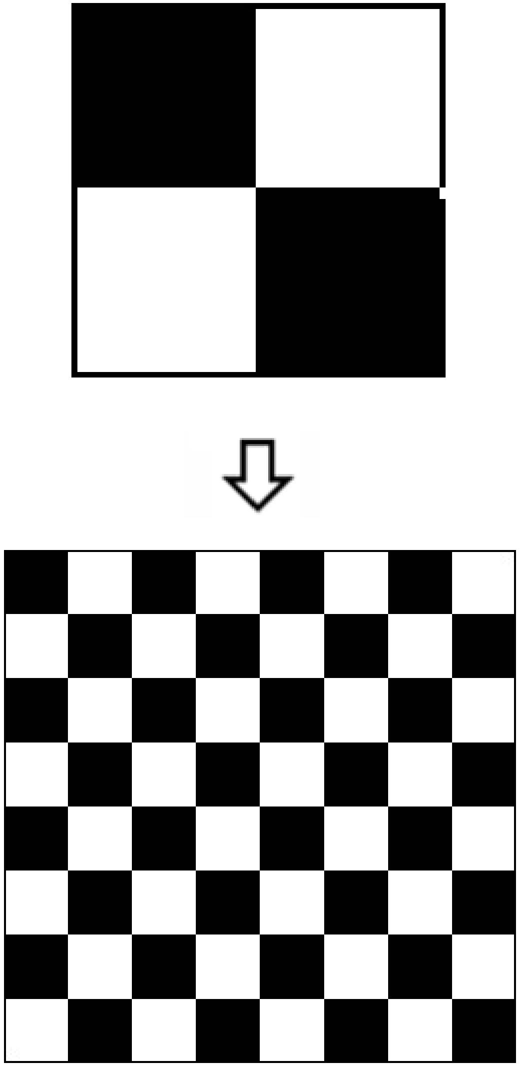
Digital Images and Formal Languages – 1st part

- *Image representation* by *formal languages*:
- Note: we used a simple trick to convert image from *example* 3 to image from *example* 4.

  Image from *example* 4 described by $\Sigma^2\{1, 2\}\Sigma^*$ is concatenation of two regular expressions:

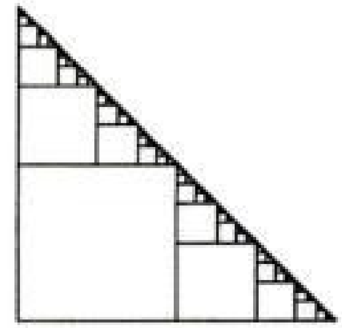
  Σ^2 and $\{1, 2\}\Sigma^*$.

  If image is described by concatenation of two languages $L = L_1 L_2$, then the image of L is always obtained by placing copies of L_2 into all squares addressed by the strings from L_1.



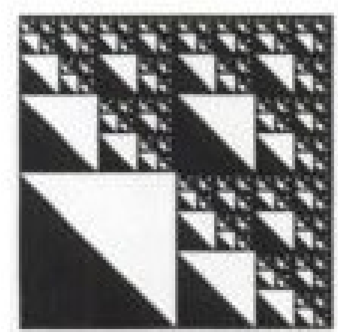
Digital Images and Formal Languages – 1st part

- *<u>Image representation</u> by <u>formal languages – fractals</u>:*

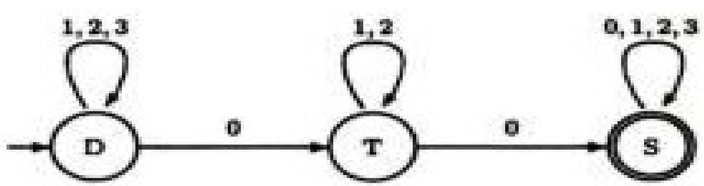


- Lets have image described by L_1 = **{1, 2}*0**, addresses of infinitely many pixels (squares):

Then lets construct new image described by $L = L_1 L_2$, where $L_2 = \Sigma^*$, so that L = **{1, 2}*0** $\mathbf{\Sigma}^*$.



Now we can construct next image described by $L' = L L_3$, where L_3 = {1, 2, 3}*0, so that L' = **{1, 2, 3}*0 {1, 2}*0** $\mathbf{\Sigma}^*$.



Digital Images and Formal Languages – 1st part

Digital Images and Formal Languages – conclusion

- All images of regular character and fractals can be perfectly (with infinite precision) described by regular expression as was shown above.
- Any image can be approximated by a regular expression (finite automata), but approximation with smaller error will require larger automata.
- In case of grayscale images the pixel values are real numbers, so multi-resolution image is then described by function ***g***: $\mathbf{\Sigma}^* \rightarrow \boldsymbol{R}$, *R* stands for real numbers.
- ***Next time: more about grayscale, image operations like zooming, filtering and compression.***

Digital Images and Formal Languages – 1st part

1. End of presentation

- **Thank you for your attention.**
- **Any questions?**