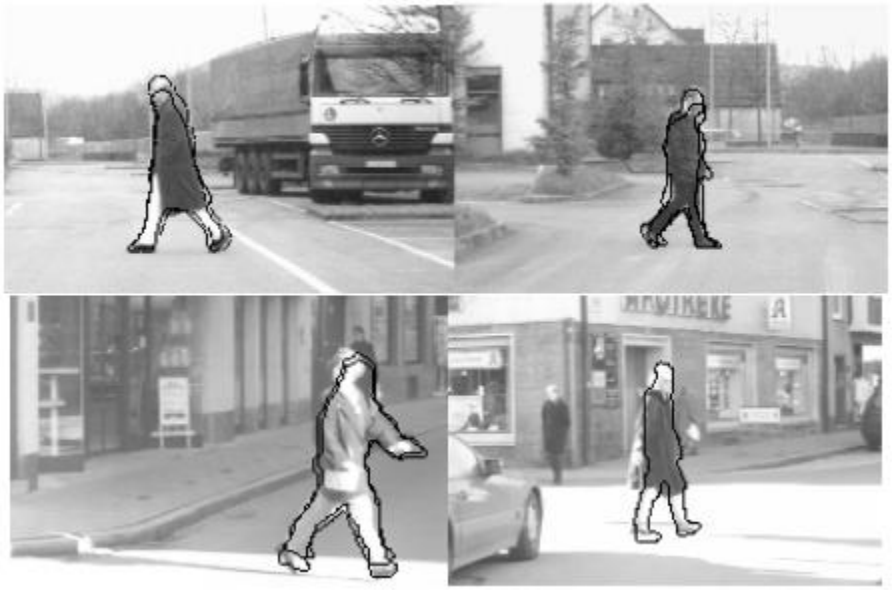
# اصول پردازش تصویر

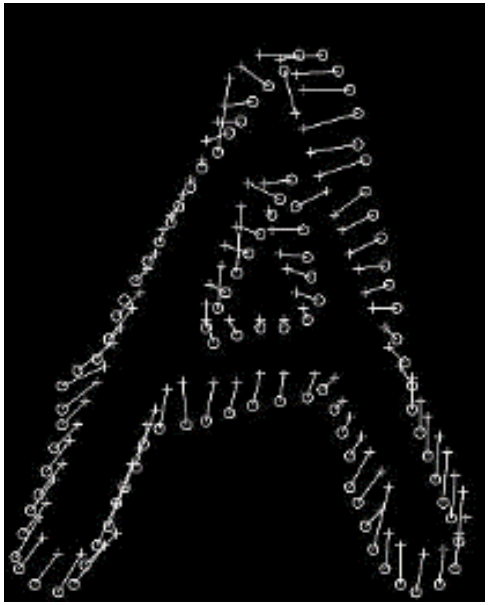
# *Principles of Image Processing*

مصطفی کمالی تبریزی

۸ دی ۱۳۹۹

جلسه بیست نهم





# Shape Matching

# Shape Matching

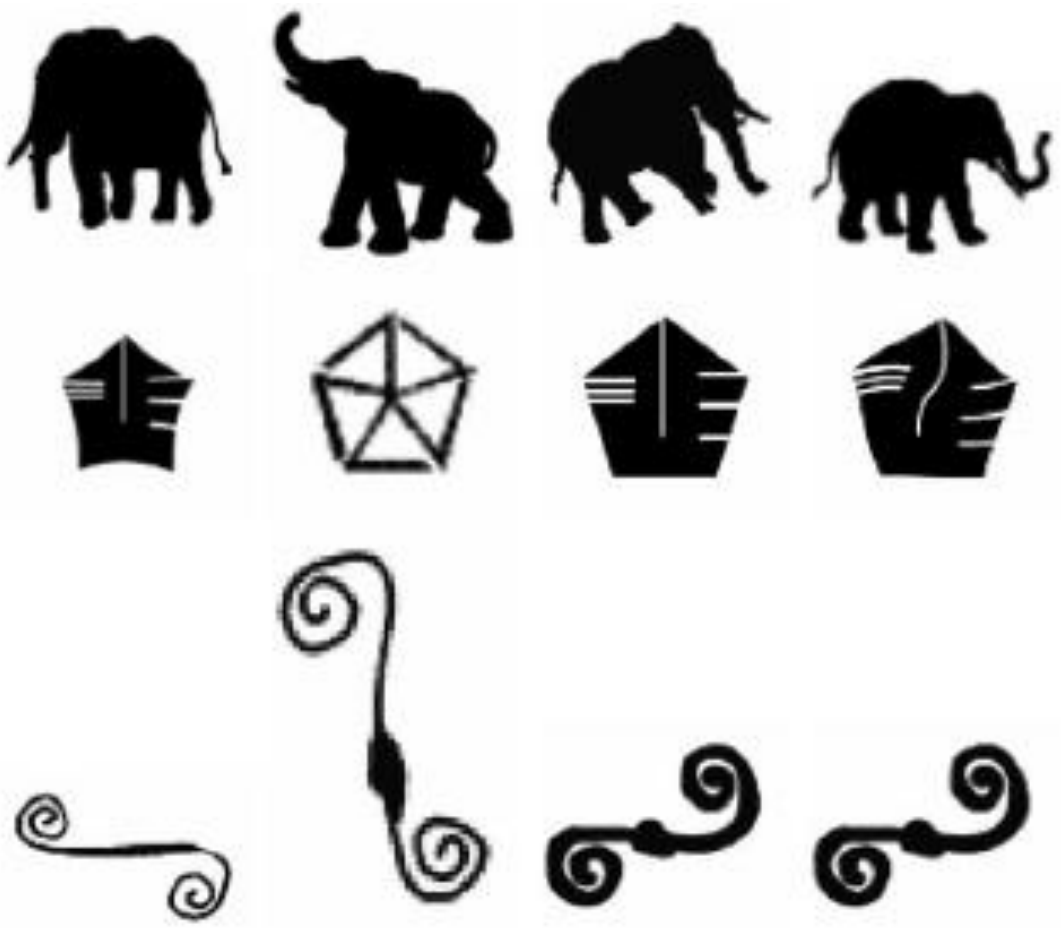


Fig. 11. Examples of shapes in the MPEG7 database for three different categories.

# Questions

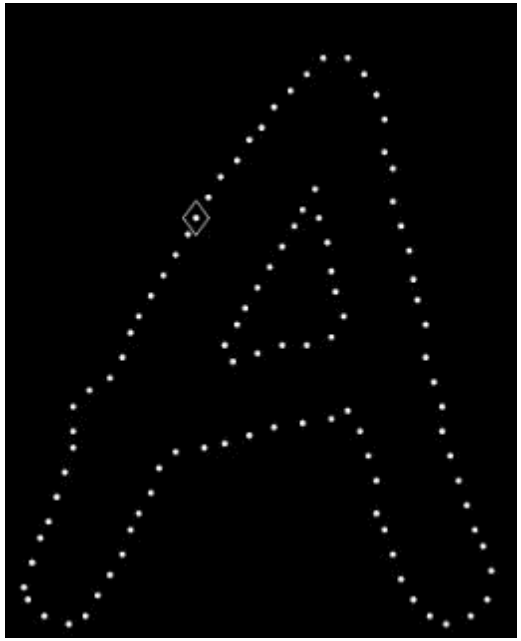
- What features?
- How to compare shapes?

# Challenging!

Fig. 1. Examples of two handwritten digits. In terms of pixel-to-pixel comparisons, these two images are quite different, but to the human observer, the shapes appear to be similar.

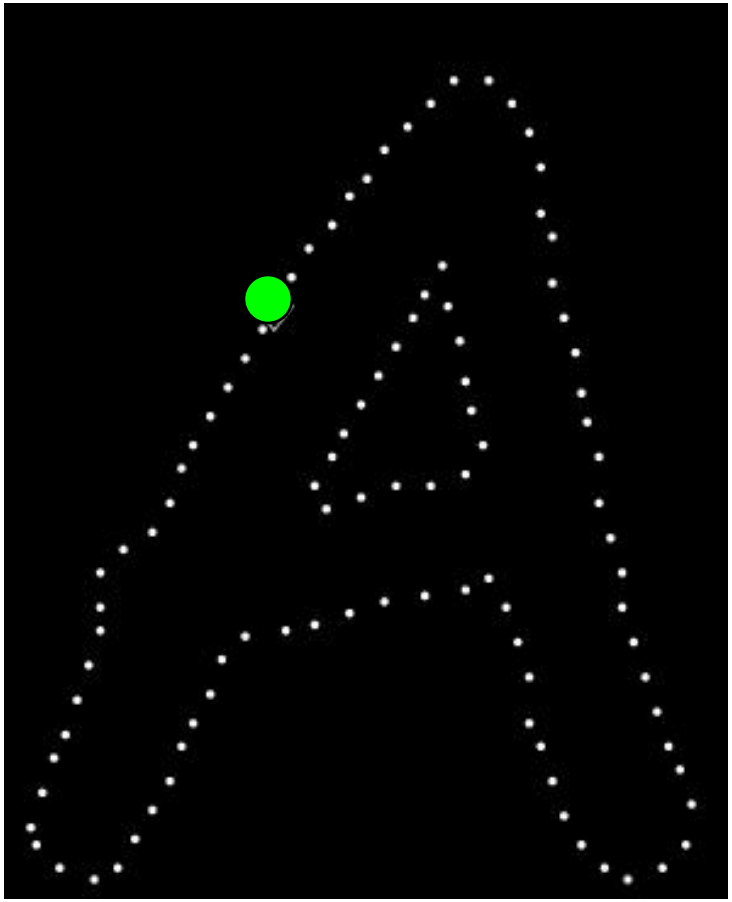
Shape context slides from Belongie et al.

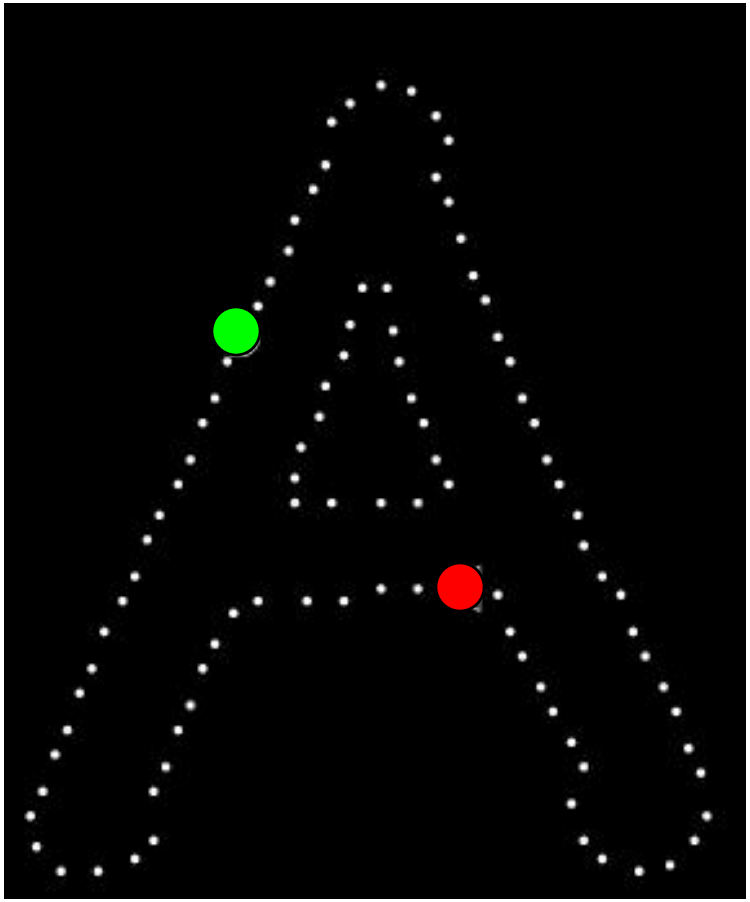
- What limitations might we have using only edge points to represent a shape?
- How descriptive is a point?



Shape context slides from Belongie et al.

# Comparing Shapes

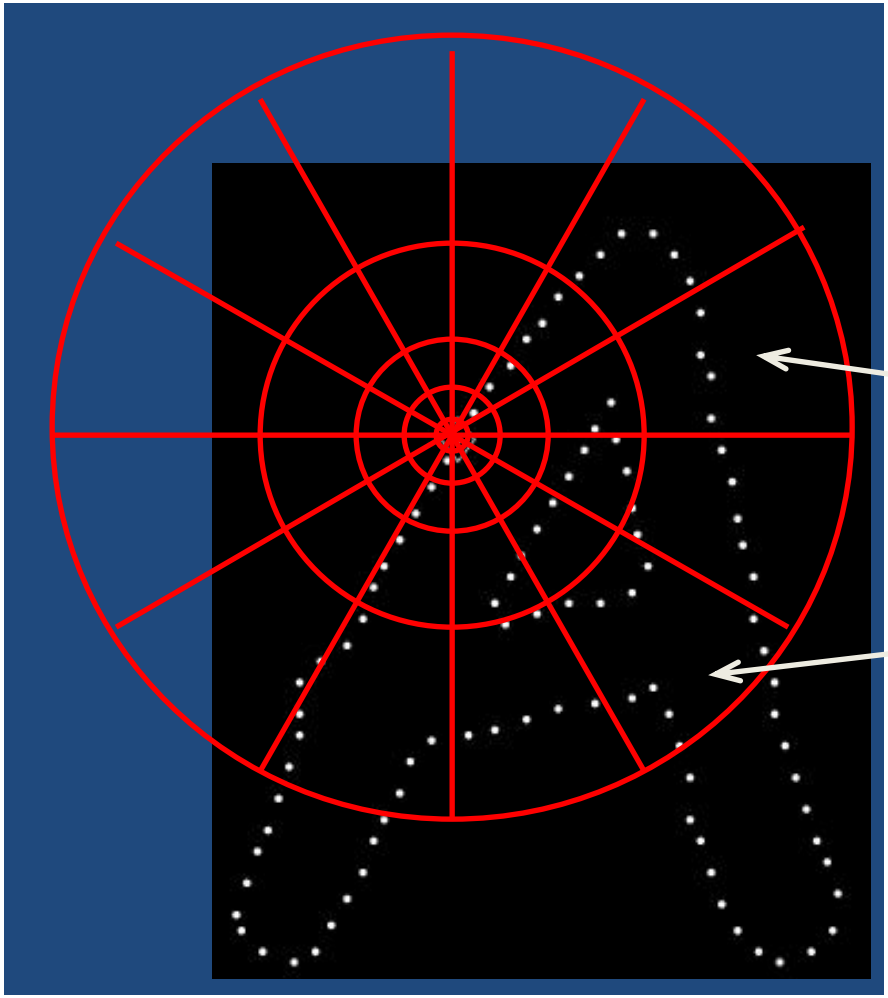




What points on these two sampled contours are most similar? How do you know?

Shape context slides from Belongie et al.

# Shape Context Descriptor



Count the number of points inside each bin, e.g.:

Count = 4

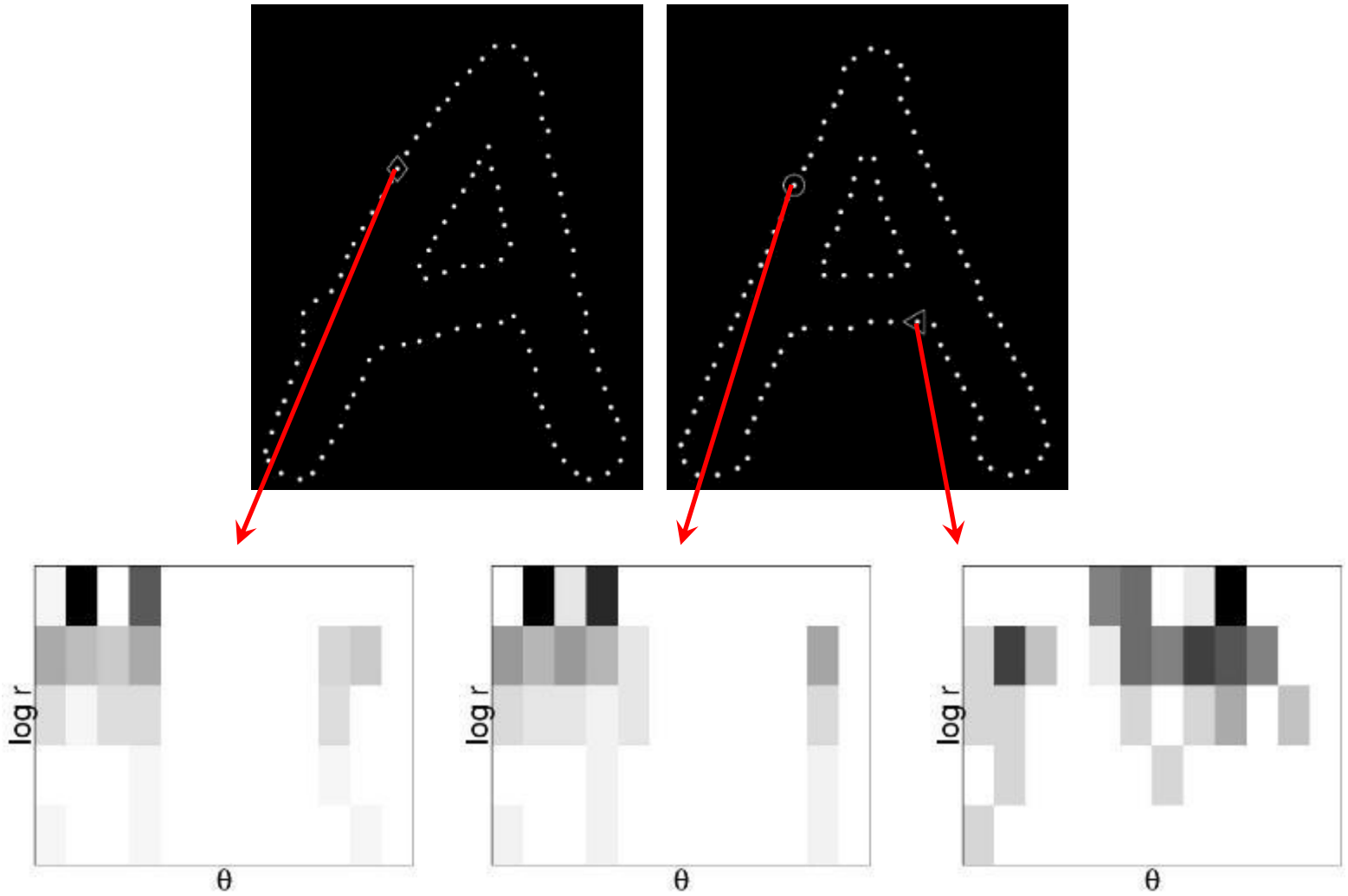
⋮

Count = 10

Compact representation of distribution of points relative to each point

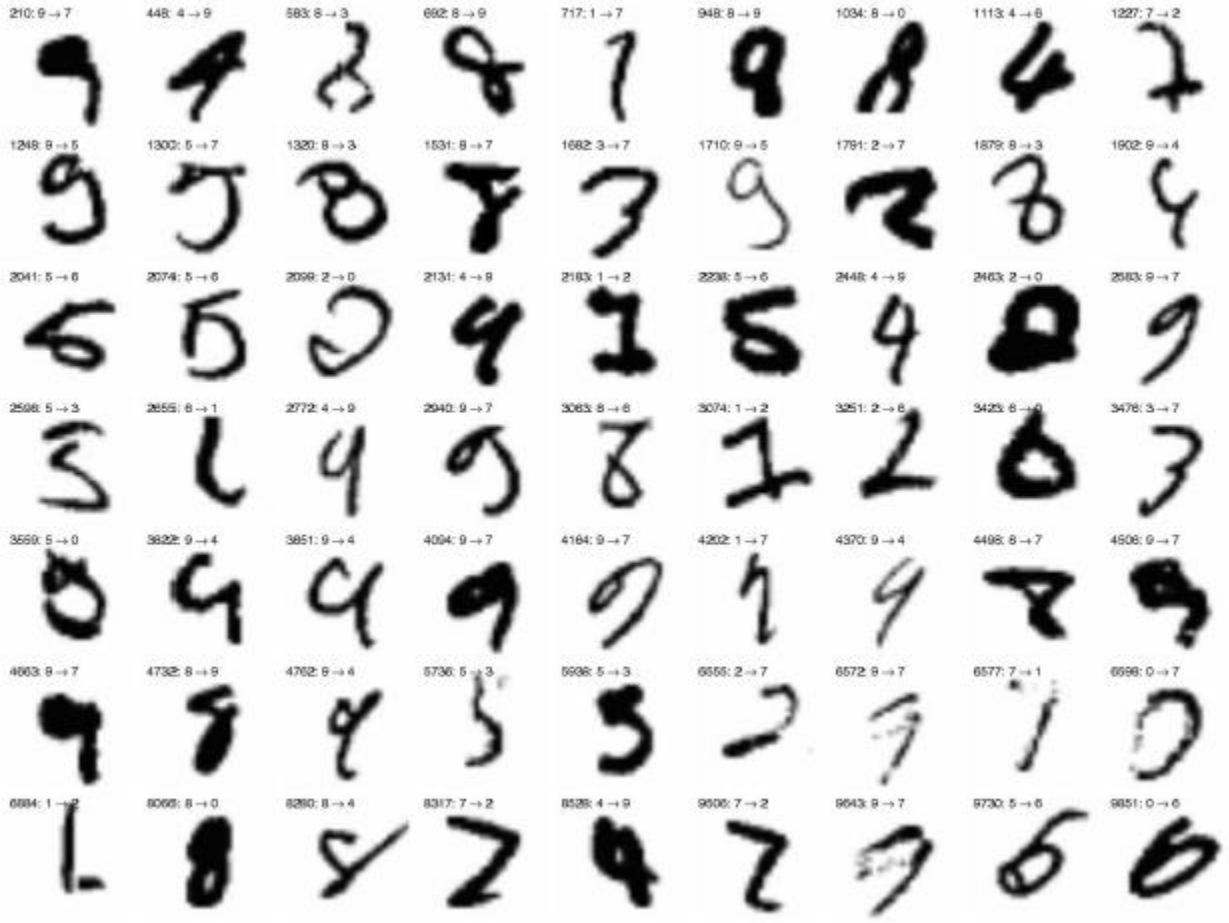
Shape context slides from Belongie et al.

# Shape Context Descriptor




Shape context slides from Belongie et al.

# Shape context matching with handwritten digits



Only errors made out of 10,000 test examples

# Shape matching application: CAPTCHA's

CAPTCHA:

"*Completely Automated Public Turing Test To Tell Computers and Humans Apart*"

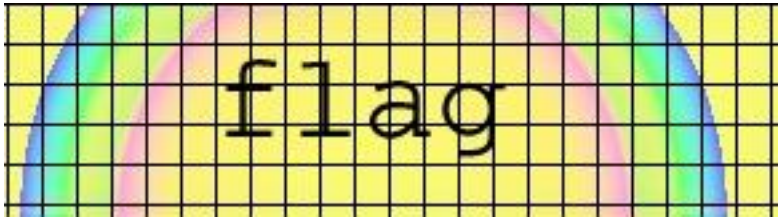
www.captcha.net

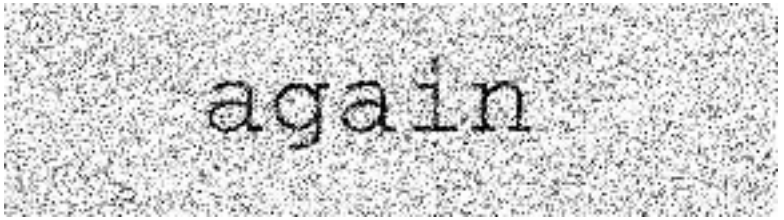
Luis von Ahn, Manuel Blum, Nicholas Hopper, and John Langford

CMU 2000

# Shape matching application: breaking a visual CAPTCHA

- Use shape matching to recognize characters, words in spite of clutter, warping, etc.



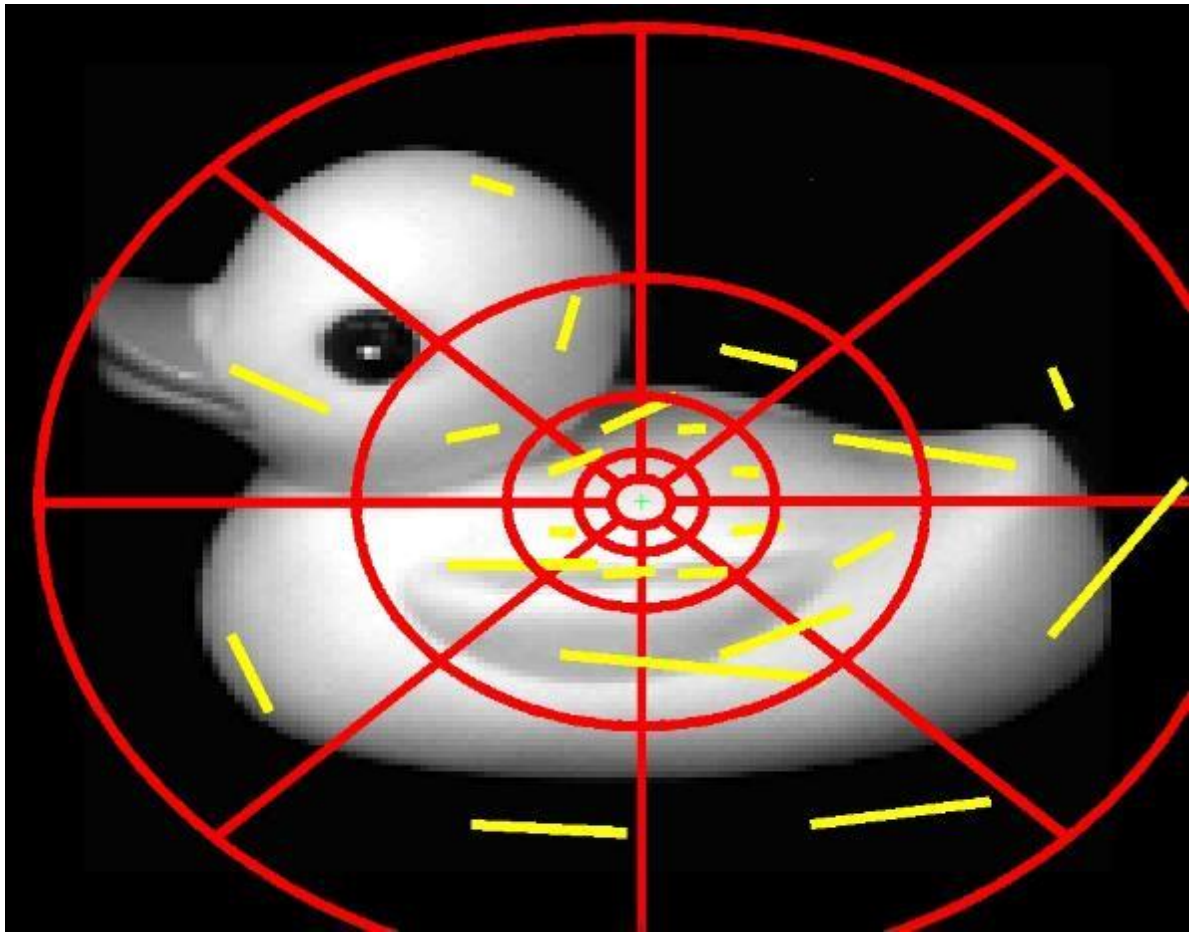





G. Mori and J. Malik, “Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA”, CVPR 2003

# Features: Generalized Shape Contexts

- Can put more than just point counts in bins
  - Oriented Energy
  - Colour info
  - Optical flow



# Fast Pruning: Representative Shape Contexts

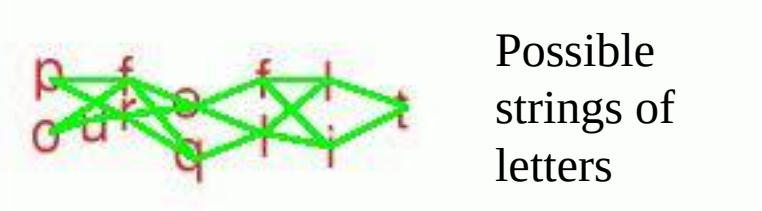

- Pick *k* points in the image at random
  - Compare to all shape contexts for all known letters
  - Vote for closely matching letters
- Keep all letters with scores under threshold

# Algorithm A: bottom-up

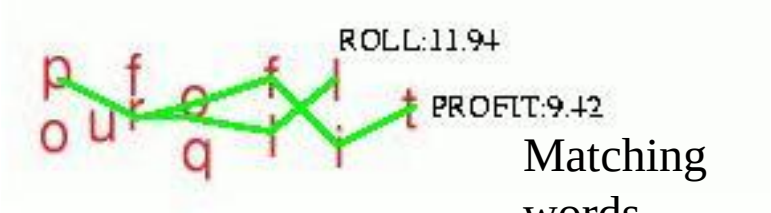
- Look for letters
  - Representative Shape Contexts
- Find pairs of letters that are "consistent"
  - Letters nearby in space
- Search for valid words
- Give scores to the words


Input

Locations of possible letters



Possible strings of letters




Matching words

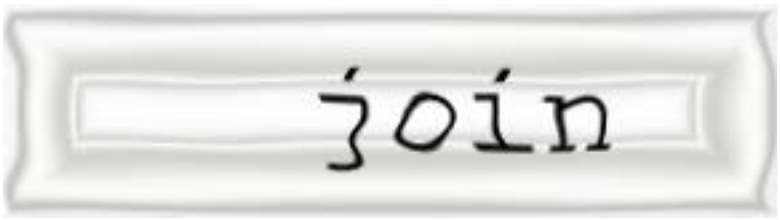
# EZ-Gimpy Results with Algorithm A

- 158 of 191 images correctly identified: 83%
  - Running time: ~10 sec. per image (MATLAB, 1 Ghz P3)


horse


spade

smile




join


canvas


here

# Seam Carving

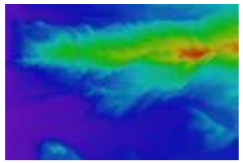
**Seam Carving for Content-Aware Image Resizing**

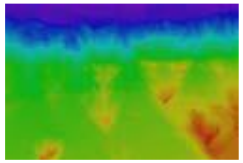
Shai Avidan
Mitsubishi Electric Research Labs

Ariel Shamir
The Interdisciplinary Center & MERL

http://www.youtube.com/watch?v=6NclJXTlugc

Slide: Alyosha Efros





# Seam Carving

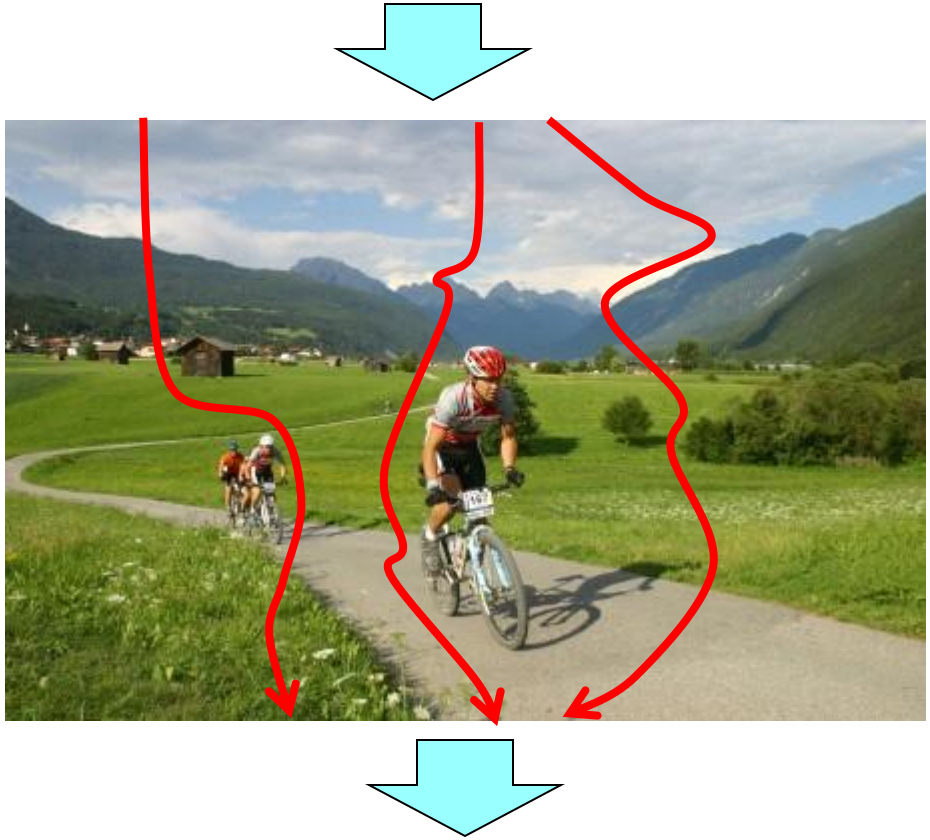
- Basic Idea: remove unimportant pixels from the image
  - Unimportant = pixels with less “energy”

$$E_1(\mathbf{I}) = |\frac{\partial}{\partial x}\mathbf{I}| + |\frac{\partial}{\partial y}\mathbf{I}|.$$

- Intuition for gradient-based energy:
  - Preserve strong contours
  - Human vision more sensitive to edges – so try remove content from smoother areas
  - Simple, enough for producing some nice results
  - See their paper for more measures they have used

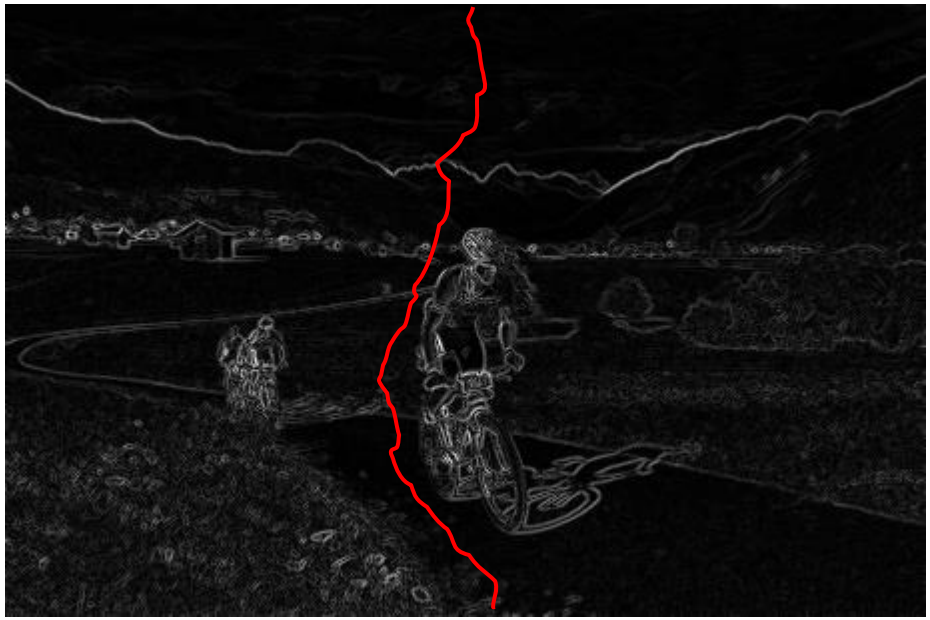
Slide: Alyosha Efros

# Finding the Seam?



Slide: Alyosha Efros

# The Optimal Seam



$$E(\mathbf{I}) = \left|\frac{\partial}{\partial x}\mathbf{I}\right| + \left|\frac{\partial}{\partial y}\mathbf{I}\right| \Rightarrow s^* = \arg\min_s E(s)$$

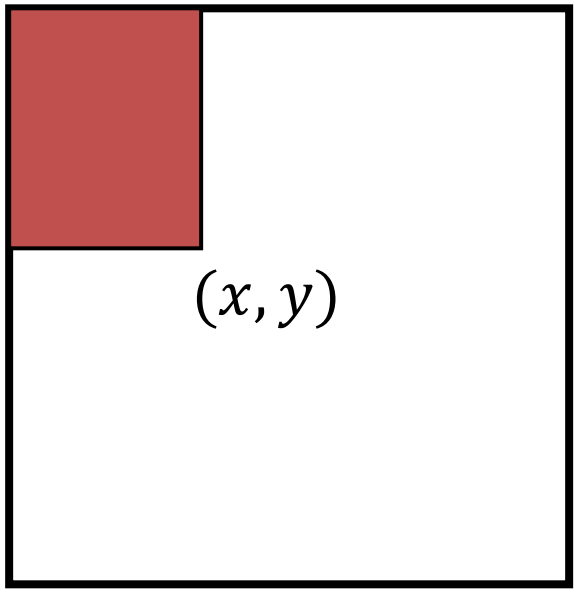
Slide: Alyosha Efros

# *Integral Images*

# *Fast computation of convolution with box filters with integral images*

- The *integral image* computes a value at each pixel $(x, y)$ that is the sum of the pixel values above and to the left of $(x, y)$, inclusive.

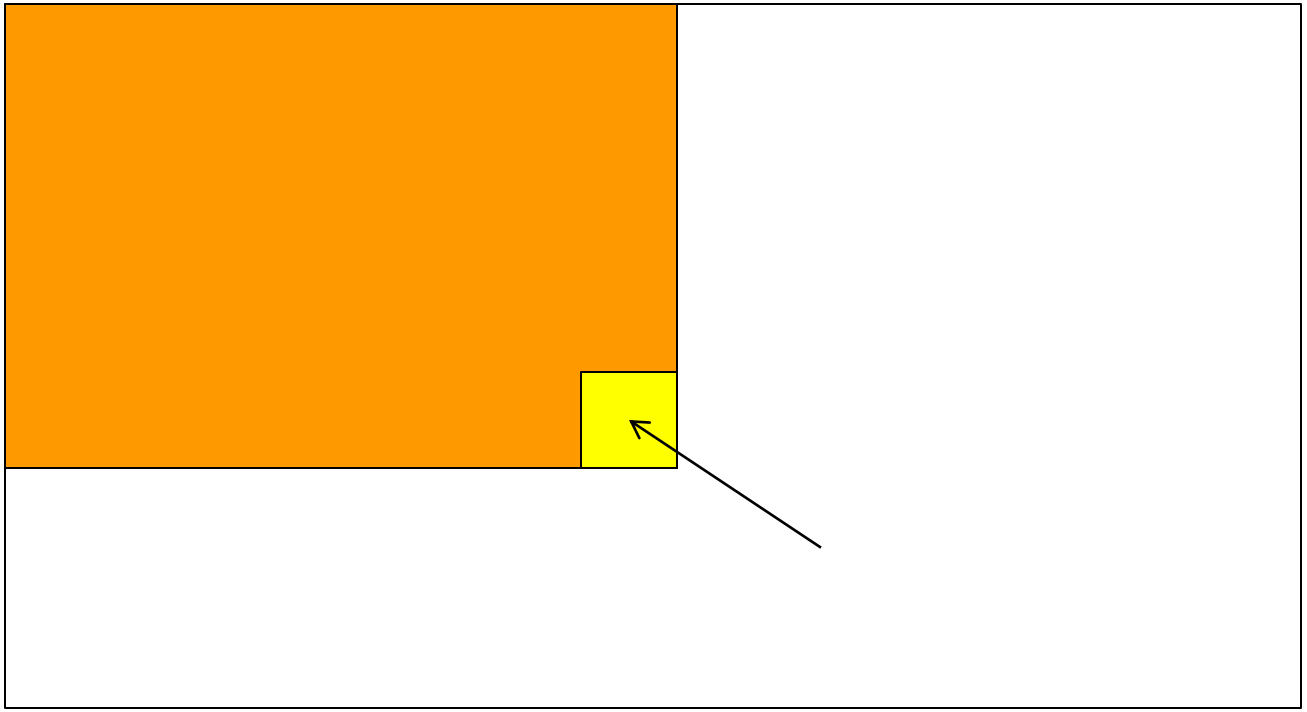
- This can quickly be computed in one pass through the image.






Slide: S. Lazebnik

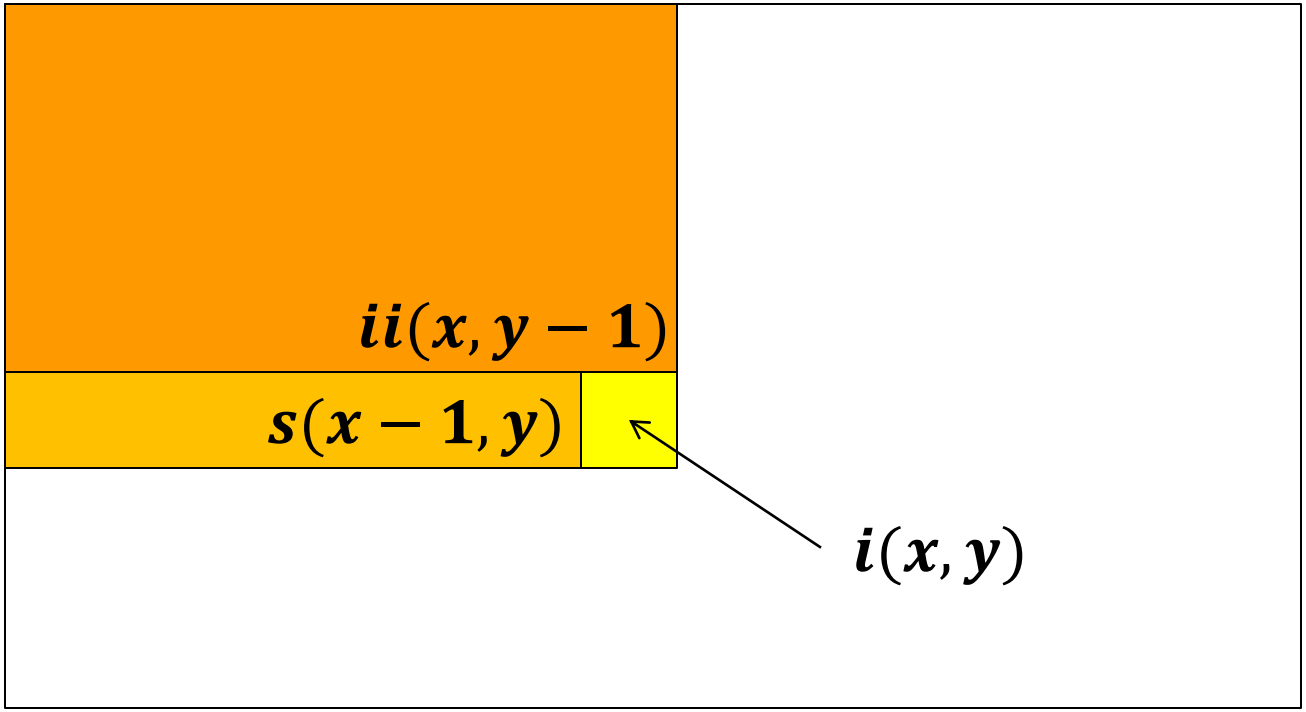
# *Computing the integral image*





Slide: S. Lazebnik

# Computing the integral image




- Cumulative row sum: $s(x, y) = s(x-1, y) + i(x, y)$
- Integral image: $ii(x, y) = ii(x, y-1) + s(x, y)$
- In MATLAB: $ii = cumsum(cumsum(double(i)), 2)$
- In Python: $ii = cumsum(cumsum(double(i), 0), 1)$

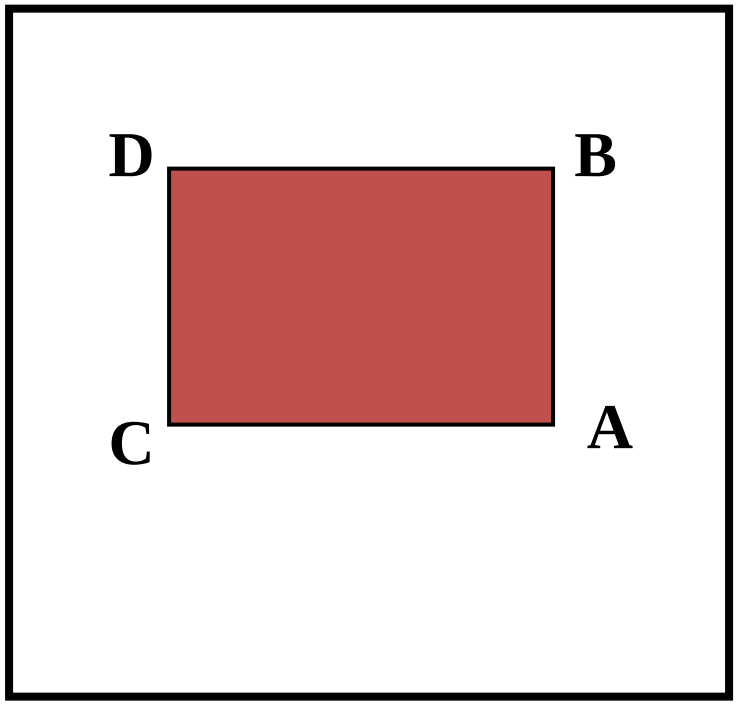


Slide: S. Lazebnik

# *Computing sum within a rectangle*

- Let A, B, C, and D be the values of the integral image at the corners of a rectangle.
- Then the sum of original image values within the rectangle can be computed as:

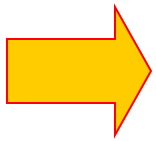
  sum = A – B – C + D
- Only 3 additions are required for any size of rectangle!

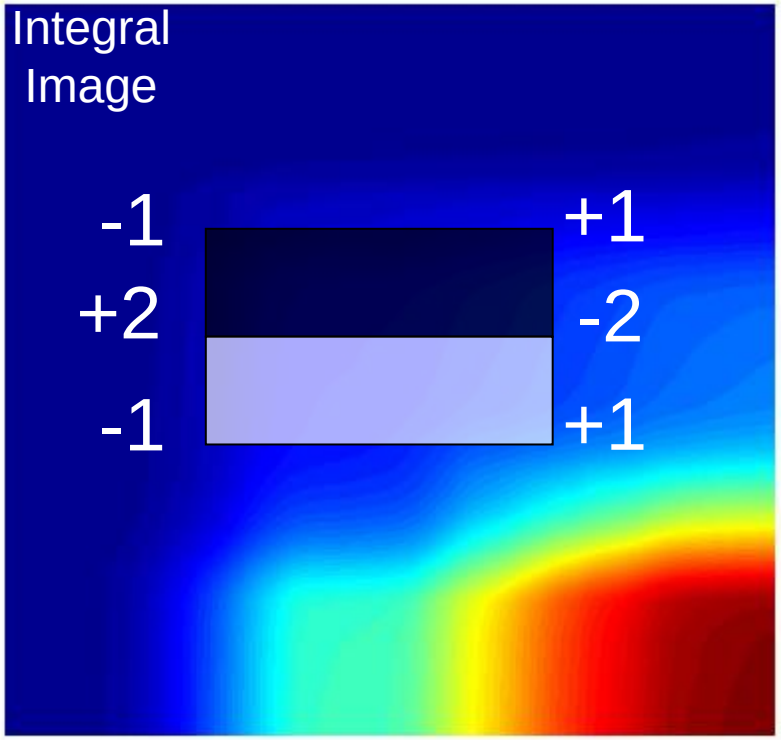





Slide: S. Lazebnik

# *Example*








Slide: S. Lazebnik