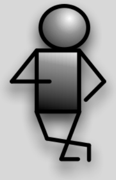
SCENE SEGMENTATION
In six easy steps

Vlad Wojcik
Dept. of Computer Science
Brock University



ABSTRACT: In this presentation we ponder the computational nature of our mental process of 3D scene construction

Starting from a pair of 2D retinal images we construct a segmented 3D scene full of objects.
We do not worry at first about the classification of these objects into known categories.
The pre-requisite to object classification is scene segmentation, i.e. transformation of our field of view into a scene full of objects, known or unknown

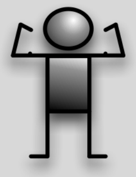
This unconscious scene segmentation process, hitherto considered rather obscure, is indeed very simple, but requires a fast and highly specialized parallel computer, if to be performed in real-time

This is a presentation of seven text and pixel processing problems, starting from an almost trivial one.
The six steps that follow lead to six problems, each obtained by a small and easily graspable generalization of previous problem. The last problem in that sequence is the scene segmentation problem

After the presentation you will fully understand why all animals have (at least) two eyes, and why human visual cortex is composed of two mirrored computers, each sitting in one of the two brain hemispheres. Furthermore, it will become clear why each of your eyes is connected to both of these computers, and why even catastrophic damage to one of these computers does not lead to blindness

All this is achieved without any recourse to biology, and offers new appreciation of the power of genetic programming and genetic algorithms

SCENE SEGMENTATION
In six easy steps



TERMS:

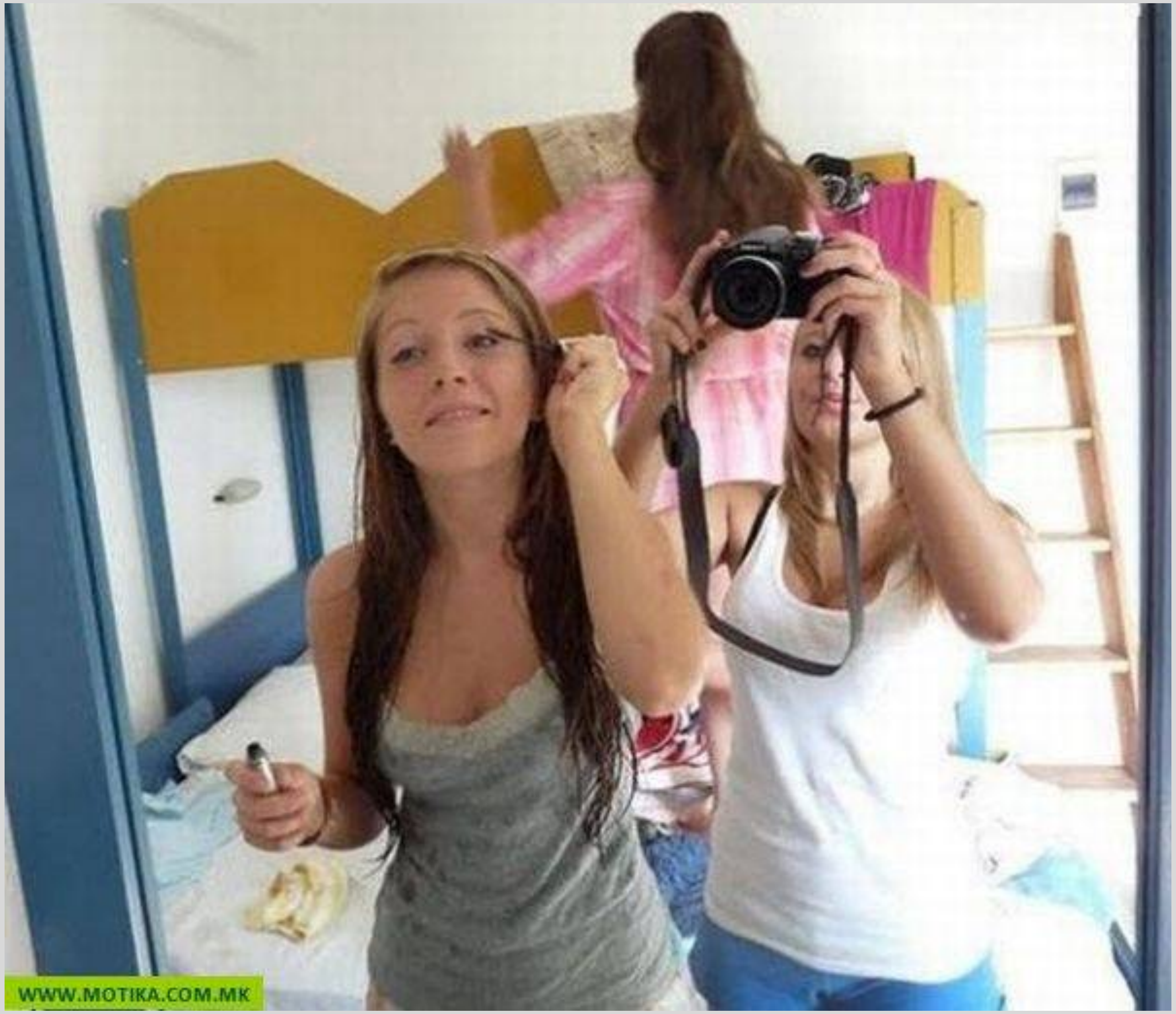
Scene: a set of information that can flow from a physical environment into a perceptual system via sensory transduction

Scene segmentation is the first step in the overall *scene understanding*. [Visual] segmentation amounts to partitioning of our field of view into a scene made of objects, known or unknown

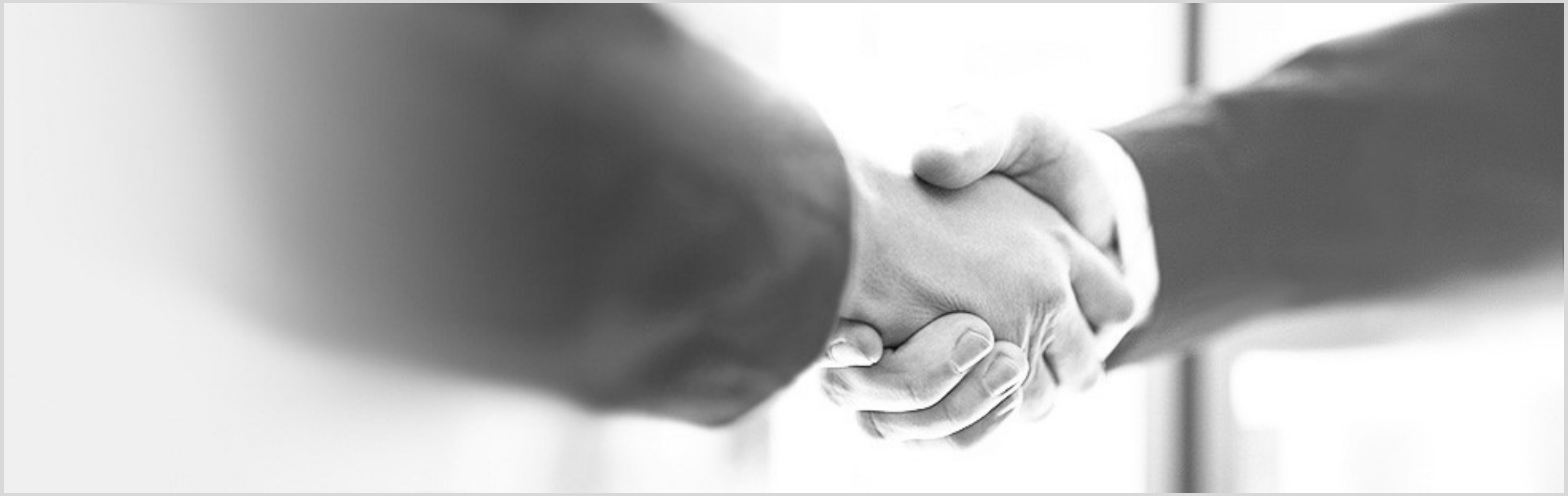
Scene understanding is the process of interpretation of the perceived *3D scene* in terms of objects, their types and their spatial relationships (relationship to the viewer, occlusion, connectedness, etc.)

3D scenes are mental models of our immediate environment, rather than just *images,* and emerge from interpretation of pairs (or n-tuples) of 2D images formed on imaging sensors (retinae, CCD sensors)

SCENE SEGMENTATION
Extra Motivation



SCENE SEGMENTATION
Issue of Depth of Field



What do you see above:

- Hands shaking? Or merely:
- Palms shaking?
- Are you sure?

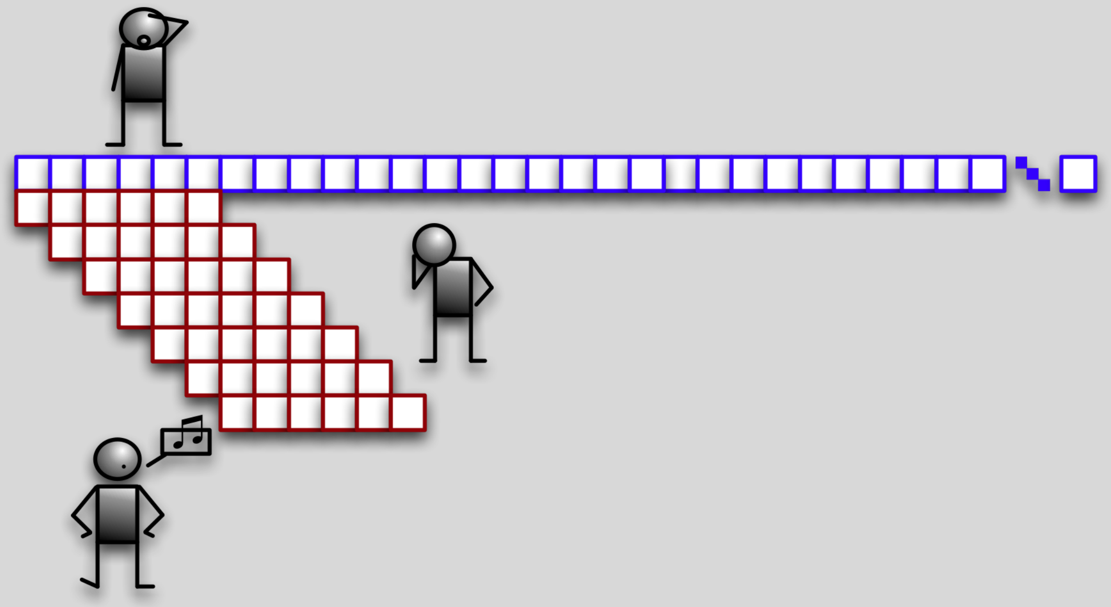
WARM-UP PROBLEM:
Text Matching

PROBLEM #0: Write a program that would read in two text strings and identify within the second string all occurrences of the first string. Observe that each instance of the substring may occur several times in the parent string. All these substring occurrences should be identified by your program.

WARM-UP PROBLEM:
Text Matching

PROBLEM #0: Write a program that would read in two text strings and identify within the second string all occurrences of the first string. Observe that each instance of the substring may occur several times in the parent string. All these substring occurrences should be identified by your program.

SOLUTION: A number of searches are needed, preferably performed in parallel:

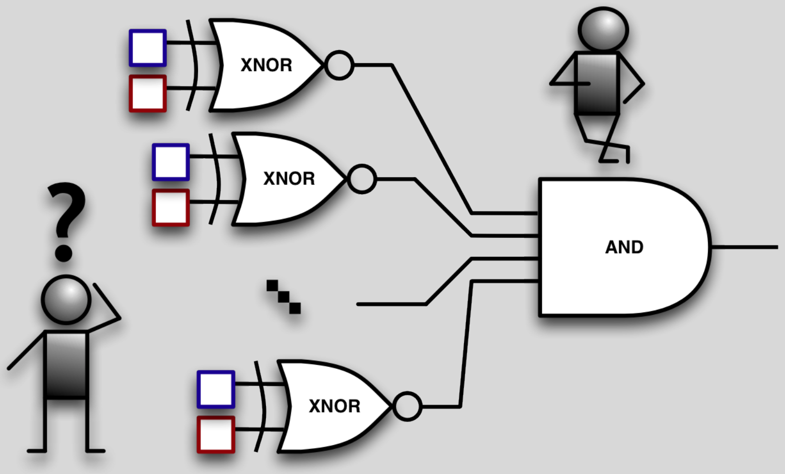


Characters are compared pairwise, and all comparisons can be accomplished in parallel by bitwise byte comparisons, viz.:

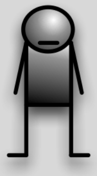
WARM -UP PROBLEM:
Text Matching Details

PROBLEM #0: Write a program that would read in two text strings and identify within the second string all occurrences of the first string. Observe that each instance of the substring may occur several times in the parent string. All these substring occurrences should be identified by your program.

SOLUTION DETAILS: Bitwise byte comparisons require particularly simple hardware and beg to be done in parallel:



Step #1 towards SCENE SEGMENTATION: Again Text Matching



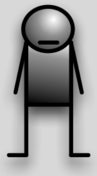
PROBLEM #1: Write a program that would read in two text strings and identify within them all occurrences of a longest common substring.

Observe that:

- The longest substring may not be unique - your program should be able to identify all instances of all such substrings in both parent strings.
- Each instance of each substring may occur several times in each parent string. All these substring occurrences should be identified by your program.

NOTE: This is a very practical problem for searches over the strings with the alphabet {A, C, G, T}

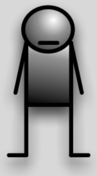
Step #2 towards SCENE SEGMENTATION: Still Text Matching



PROBLEM #2: The problem is identical to Problem 1, but now you allow n misspellings in the substring, where n = 0 , 1, 2, 3 .. What is the maximum value of n that makes sense?

NOTE: This (again) is a very practical problem for searches over the strings with the alphabet {A, C, G, T}

Step #3 towards SCENE SEGMENTATION:
B&W Pixel Matching



PROBLEM #3: The problem is similar to Problem 2, but now we scan two rows of B&W pixels, with the input alphabet of the two byte (pixel) arrays consists of shades of gray in the range 0 .. 255, rather than text characters.

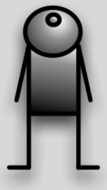
Pixel match criteria:

- Two pixels being compared are considered matching, if their intensity values differ by at most m gray levels, where m = 0, 1, 2, 3 .. What is the maximum value of m that makes sense?

- You still allow n misspellings in the substring of pixels, where n = 0 , 1, 2, 3 ..
  What is the maximum value of n that makes sense?

Step #4 towards SCENE SEGMENTATION: Colour Pixel Matching

PROBLEM #4: The problem is similar to Problem 3, but now the input consists of two rows of colour pixels, i.e. six byte arrays representing shades of gray in the range 0 .. 255. First three arrays contain Red, Green, Blue pixel intensities of first input row of pixels, the remaining three arrays contain RGB intensity values of the second row of pixels.



The pixels are to be compared by colour pair wise: Red with Red, Green with Green, Blue with Blue.

Single pixel-pair match criterion (many variants are possible):

Two pixels match if:

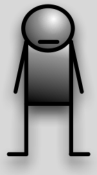
1. The sum of all three differences in RGB intensity levels lies within m gray levels, where m = 0, 1, 2, 3 ..
2. The sum of two largest differences in RGB intensity levels lies within m gray levels, where m = 0, 1, 2, 3 ..
3. The largest difference in RGB intensity levels lies within m gray levels, where m = 0, 1, 2, 3 ..
4. All three intensity levels (RGB) are identical, i.e. m = 0 in all cases above.

Which variant makes most sense? What is the maximum value of m that makes sense in each case?

Pixel pattern match criterion:

Two pixel substrings match, if they differ in no more than n pixel pairs, where n = 0 , 1, 2, 3 .. What is the maximum value of n that makes sense?

Step #5 towards SCENE SEGMENTATION: B&W Image Region Matching



PROBLEM #5: The problem is similar to Problem 3, with the input alphabet of the two B&W pixel arrays gray intensities in the range 0 .. 255.

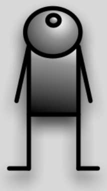
However, we consider now two B&W images as two X×Y arrays of pixels. We are searching now for the largest common 2D sub region in the two B&W images. We measure the size of the sub regions in terms of the number of their constituent pixels.

Match criteria:

- Two gray levels in the pixels being compared are considered matching, if their values differ by at most m gray levels, where m = 0, 1, 2, 3 .. What is the maximum value of m that makes sense?
- You still allow n misspellings in the substring, where n = 0 , 1, 2, 3 .. What is the maximum value of n that makes sense?

Step #6 (At last!): SCENE SEGMENTATION: Colour Image Region Matching

PROBLEM #6: The problem is similar to Problems 4 and 5, but now the input consists two colour images. Each image is represented as an X×Y array of colour pixels, or, alternatively, as three B&W arrays of gray pixels, representing RGB intensity levels.



We are searching now for the largest common 2D sub region in the two B&W images. We measure the size of the sub regions in terms of the number of their constituent pixels.

Single pixel-pair match criterion (many variants are possible):

Two pixels match if:

1. The sum of all three differences in RGB intensity levels lies within m gray levels, where m = 0, 1, 2, 3 ..
2. The sum of two largest differences in RGB intensity levels lies within m gray levels, where m = 0, 1, 2, 3 ..
3. The largest difference in RGB intensity levels lies within m gray levels, where m = 0, 1, 2, 3 ..
4. All three intensity levels (RGB) are identical, i.e. m = 0 in all cases above.

Which variant makes most sense? What is the maximum value of m that makes sense in each case?

Region match criterion:

Two pixel regions match, if they differ in no more than n pixel pairs, where n = 0 , 1, 2, 3 .. What is the maximum value of n that makes sense?

ANY QUESTIONS?