

Design of Multimedia Applications: Video Shot Detection, Annotation, & Retrieval

October 24, 2014

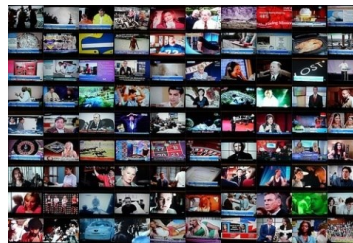
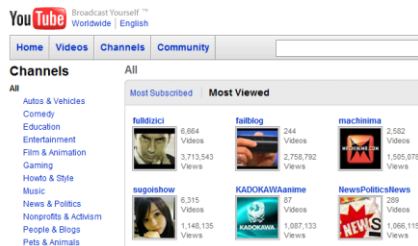
Sebastiaan Van Leuven, Niels Van Kets,
Viktor Slavkovikj, Frédéric Godin

Lectures and Exercises

- Two introductory lectures
 - topics
 - error correction in digital video (26 September)
 - video shot detection, annotation, & retrieval (24 October)
 - complementary to the theory lectures
 - background information
- Exercises
 - under supervision in Technicum: RTB Les 2.14
 - every Monday, from 16h00 till 18h45
 - attendance not mandatory

Introduction (1/4)

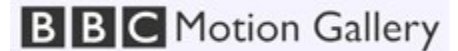
- Increasing **consumption** of online video content
 - thanks to easy-to-use (mobile) devices & online services
 - thanks to cheap storage and bandwidth
 - thanks to an increasing number of people going online
- Increasing **availability** of online video content
 - thanks to the digitization of professional video archives
 - thanks to the popularity of user-generated video content



Introduction (2/4)

- Some statistics

- professional video content



- BBC Motion Gallery (as of January 2009)

- contains over 2.5 million hours of video content
 - with some video content dating back 60 years in time

- user-generated video content

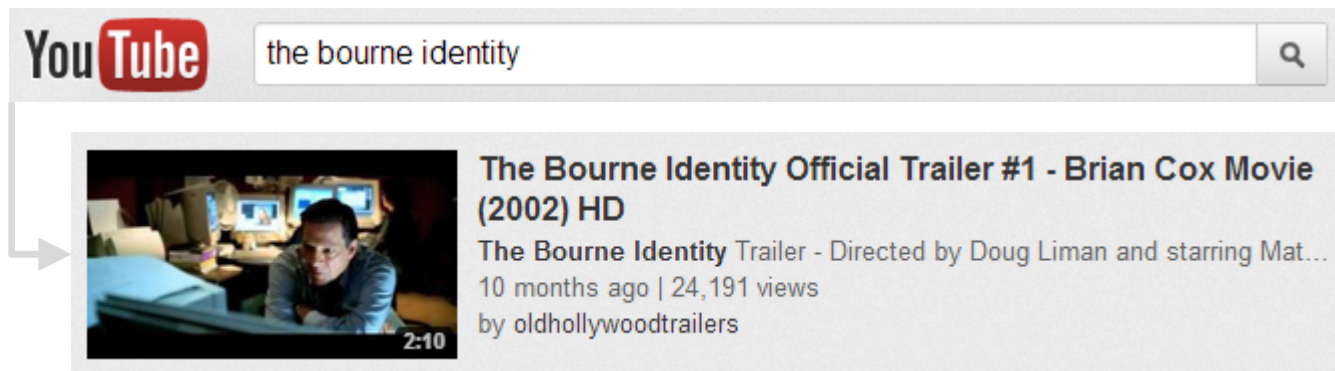


- YouTube (as of October 2013)

- people watch over 6 billion hours of video each month
 - people upload 100 hours of video each minute

Introduction (3/4)

- Problem: digital video overload (infobesity)
 - our ability to organize video clips does not keep up with our ability to create video clips
- Bottleneck: manual annotation
 - to organize video clips, we need to add descriptions to video clips ourselves (cf. text-based retrieval of video)



Introduction (4/4)

- Solution: automatic video content understanding
 - identification of the temporal structure of video clips
 - scenes and shots
 - use of machine learning (e.g., SVM, deep learning) to translate binary pixel values into text labels (tags)
 - e.g., 'people', 'building', 'news', and so on



Shot 1: 'surveillance camera'



Shot 2: 'Jason Bourne', 'people'

Goal Lab Session

- To give a better insight into the problem of organizing large collections of **video clips** by
 - implementing and evaluating a number of state-of-the-art algorithms for automatic **video shot detection**
 - and by subsequently integrating these algorithms into a **DirectShow-based application** that offers support for the **manual annotation and tag-based retrieval of video shots**

Outline

- Introduction

- Techniques for video shot detection
 - pixel-wise comparison
 - selective pixel comparison
 - block-based comparison
 - motion estimation
 - histogram-based comparison
- Performance evaluation

algorithmic
development

- DirectShow

application
development

- Assignment

Video Shots (1/2)

- Video shot
 - set of consecutive pictures
 - taken by a single camera
 - represent a continuous action in time and space
- Video shot detection
 - automatic detection of the start (and end) of shots
 - also referred to as
 - shot boundary detection
 - shot transition detection

Video Shots (2/2)

- Video shots
 - cuts: instantaneous transition between two shots



- dissolves: gradual transition between two shots



- fades: gradual transition between a shot and an image



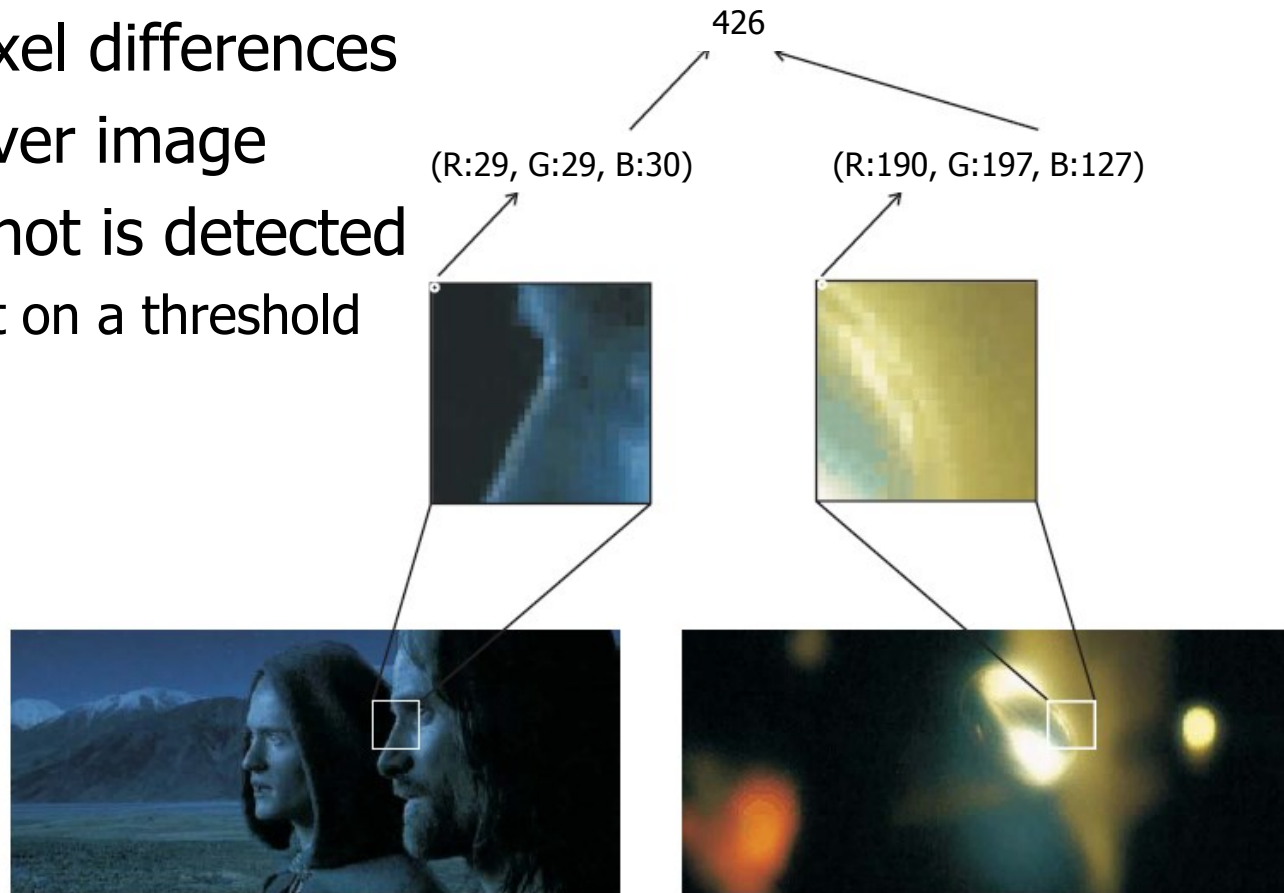
➡ to detect shots, compute and interpret frame differences

Outline

- Introduction
- **Techniques for video shot detection**
 - pixel-wise comparison
 - selective pixel comparison
 - block-based comparison
 - motion estimation
 - histogram-based comparison
- Performance evaluation
- DirectShow
- Assignment

Pixel-Wise Comparison

- Approach
 - calculate pixel differences
 - averaged over image
 - if large, a shot is detected
 - dependent on a threshold

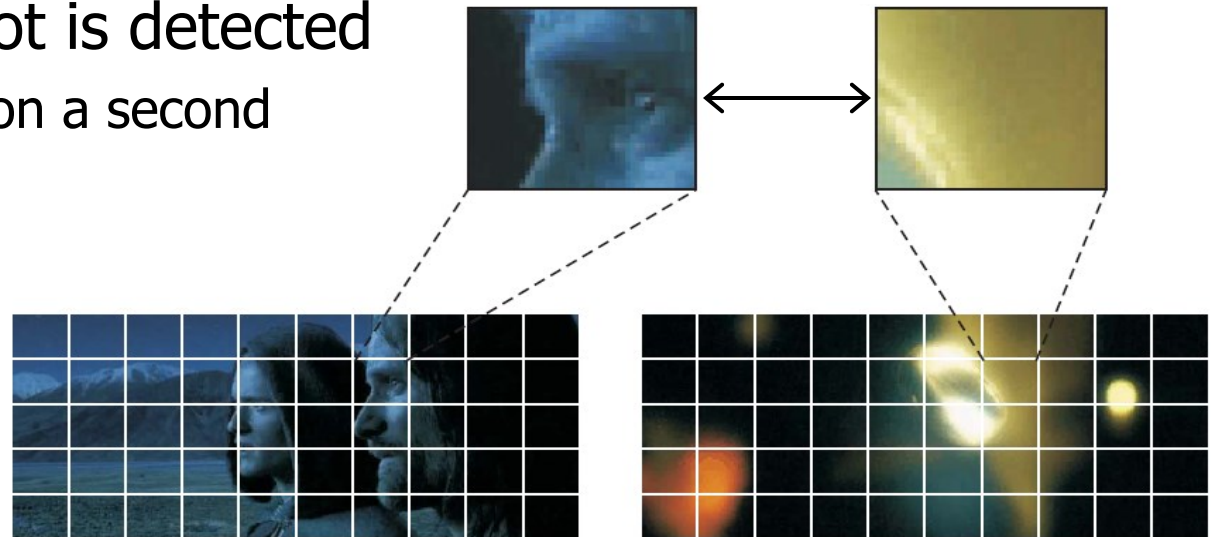


Selective Pixel Difference

- Approach
 - only use large pixel differences
 - dependent on a first threshold
 - **count** the number of large pixel differences
 - average over the image
 - if large, a shot is detected
 - dependent on a second threshold
- Prevents that lots of small differences have an influence on the detection of shots
 - small camera or object movement may have a substantial influence on the pixel differences

Block-based Comparison

- Approach
 - calculate block differences
 - count the number of **blocks** that have a large difference
 - dependent on a first threshold
 - if large, a shot is detected
 - dependent on a second threshold

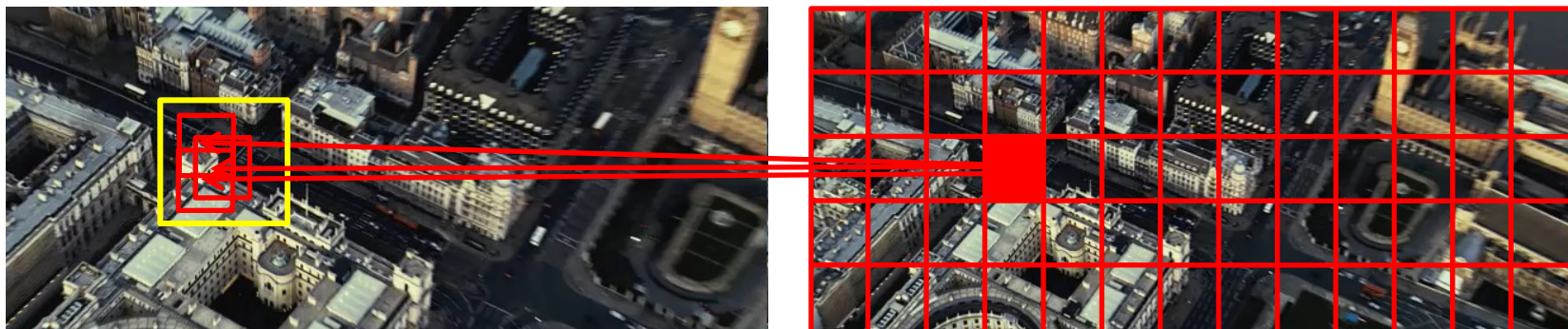


Outline

- Introduction
- Techniques for video shot detection
 - pixel-wise comparison
 - selective pixel comparison
 - block-based comparison
 - **motion estimation**
 - histogram-based comparison
- Performance evaluation
- DirectShow
- Assignment

Motion Estimation (1/3)

- Approach
 - divide current image into blocks
 - search for best matching block in previous image
 - search window centered around block position
 - calculate block difference
 - best match minimizes the block difference

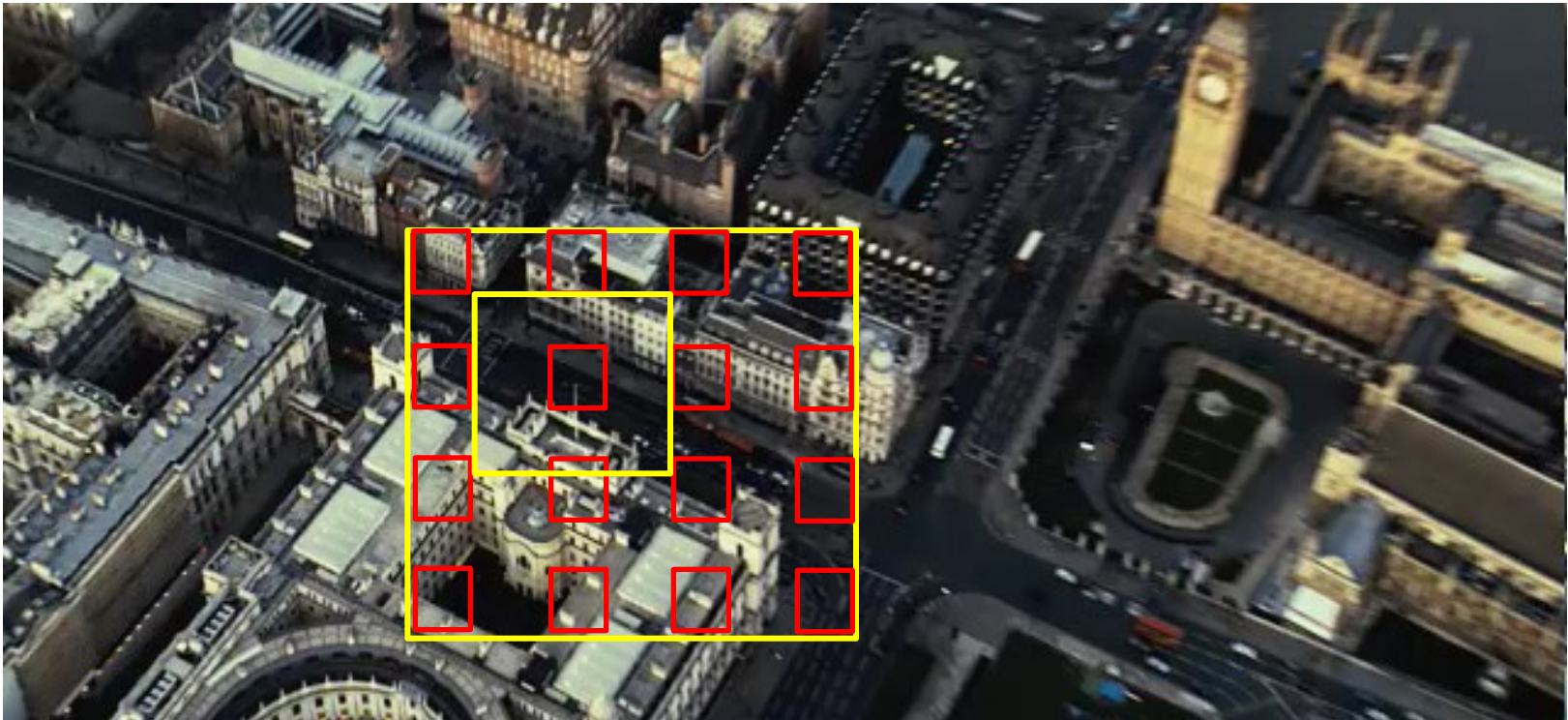


Motion Estimation (2/3)

- Approach
 - sum the matched block differences to find the overall image difference
 - if large, a shot is detected
 - dependent on a threshold
- Problem: exhaustive search is slow for large windows
 - coarse-to-fine search
 - first look at a subset of blocks in the search window
 - find the best match
 - look at blocks close to this best match
 - find a better match

Motion Estimation (3/3)

- Coarse-to-fine motion estimation

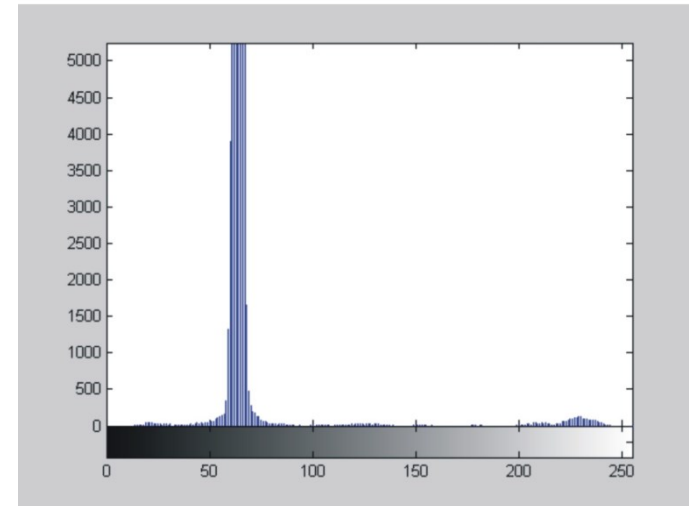


Outline

- Introduction
- Techniques for video shot detection
 - pixel-wise comparison
 - selective pixel comparison
 - block-based comparison
 - motion estimation
 - **histogram-based comparison**
- Performance evaluation
- DirectShow
- Assignment

Histogram-based Comparison (1/4)

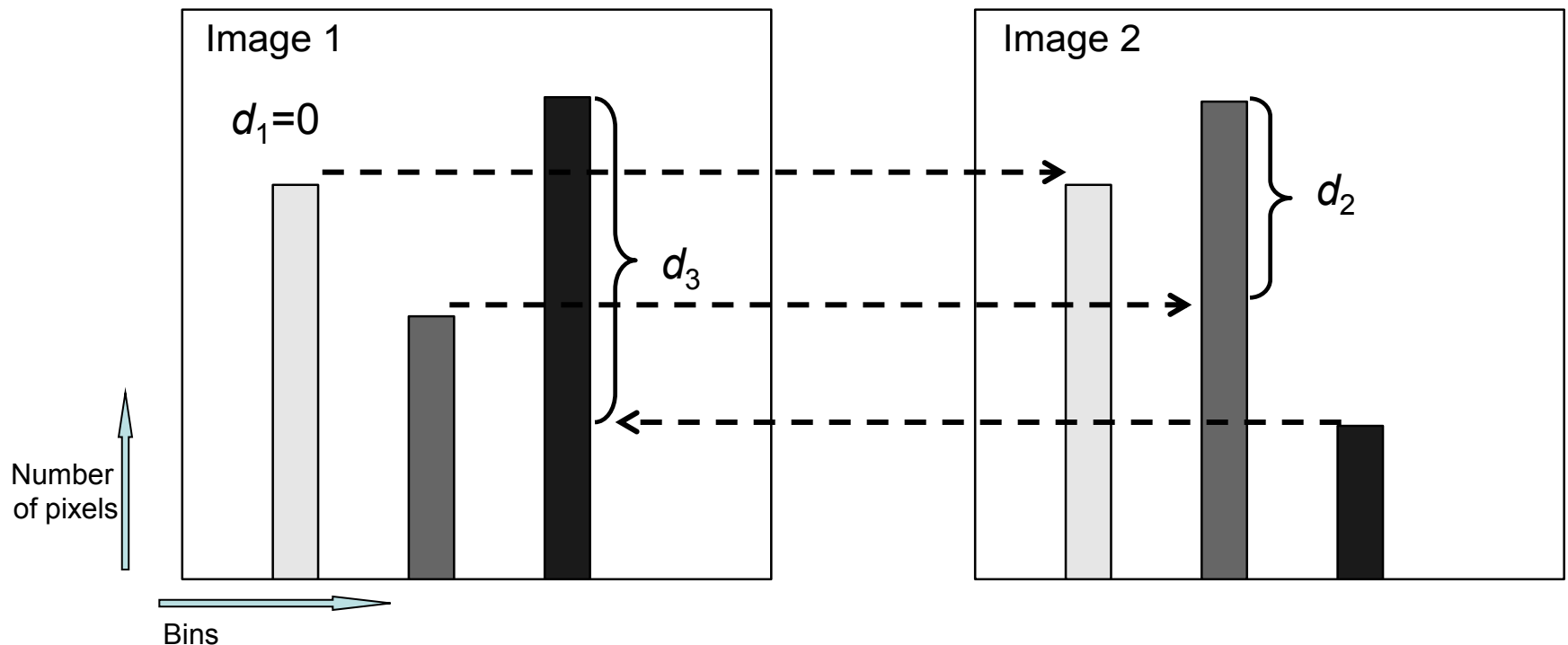
- Approach
 - count the number of pixels that have a certain color
 - image difference = histogram difference



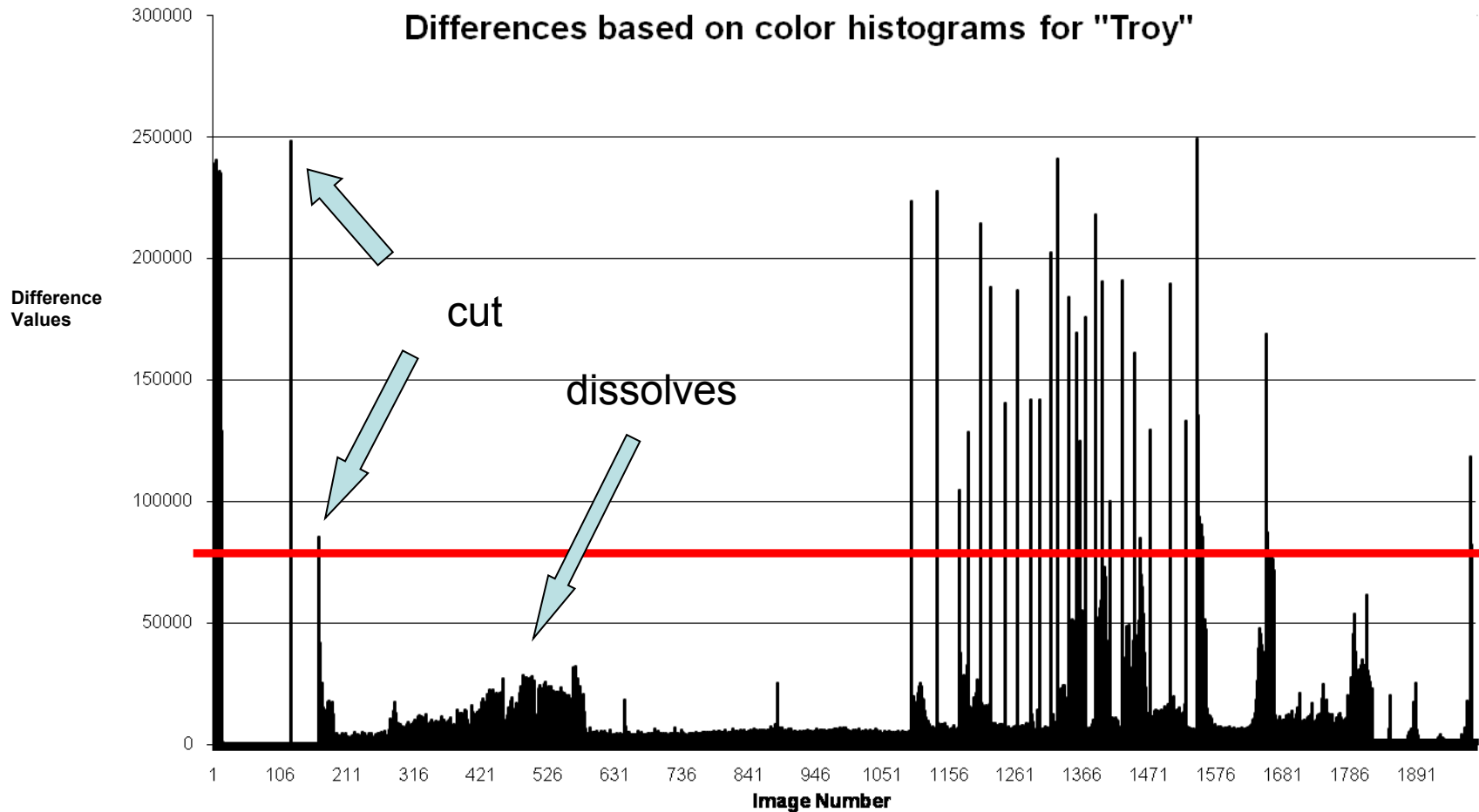
- use of bins allows reducing the complexity
 - groups pixels that have a color within a certain range

Histogram-based Comparison (2/4)

- Histogram difference
 - for each bin, compute the difference
 - many distance metrics possible: L_1 , L_2 , EMD, SQFD, ...

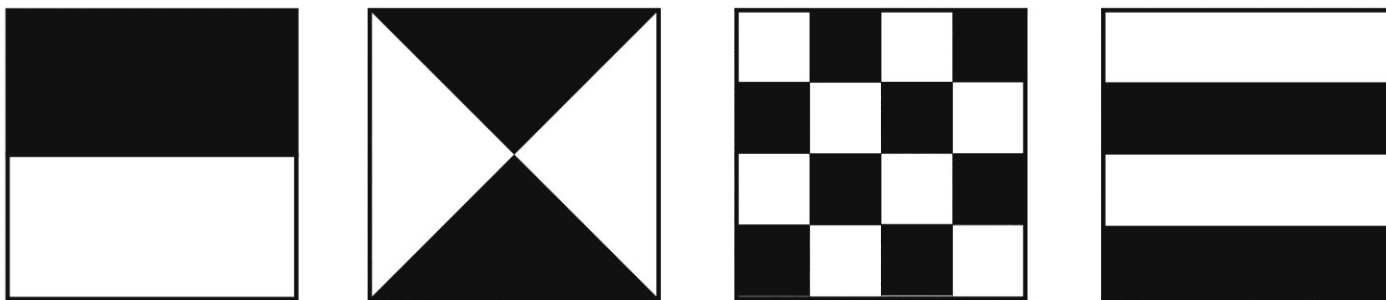


Histogram-based Comparison (3/4)



Histogram-based Comparison (4/4)

- Global histogram comparison
 - problem: different images can have similar histograms



- solution: local histogram comparison
 - calculate histograms for regions in the image
 - takes into account discriminative spatial information
 - sum histogram differences of corresponding regions in consecutive frames
 - often used in practice (and in other application domains)

Outline

- Introduction
- Techniques for video shot detection
 - pixel comparison
 - selective pixel comparison
 - block-based comparison
 - motion estimation
 - histogram comparison
- Performance evaluation
- DirectShow
- Assignment

Evaluation (1/4)

- Outcome of a particular shot detection technique
 - shot detected: referred to as a “positive”
 - no shot detected: referred to as a “negative”
 - “positives” and “negatives” can either be true or false
- Verification by means of a ground truth
 - allows comparing the output of a shot detection technique with what is in the video clip in reality
 - manually created and trustworthy
 - use of human computing (e.g., Amazon Mechanical Turk)
 - used in many other application domains
 - e.g., face recognition, video copy detection, ...

Evaluation (2/4)

Output method		Ground truth
Pos	← True Positive →	Pos
Neg	← True Negative →	Neg
Neg	← True Negative →	Neg
Neg	← True Negative →	Neg
Pos	← False Positive →	Neg
Neg	← False Negative →	Neg
Neg	← False Negative →	Pos
...		...

- Goal
 - maximize *true* positives and *true* negatives
 - minimize *false* positives and *false* negatives

Evaluation (3/4)

- Precision and recall
 - precision: percentage of correctly detected shots
 - recall: percentage of all real shots detected
 - sometimes also combined into an F_1 -score (harmonic mean)

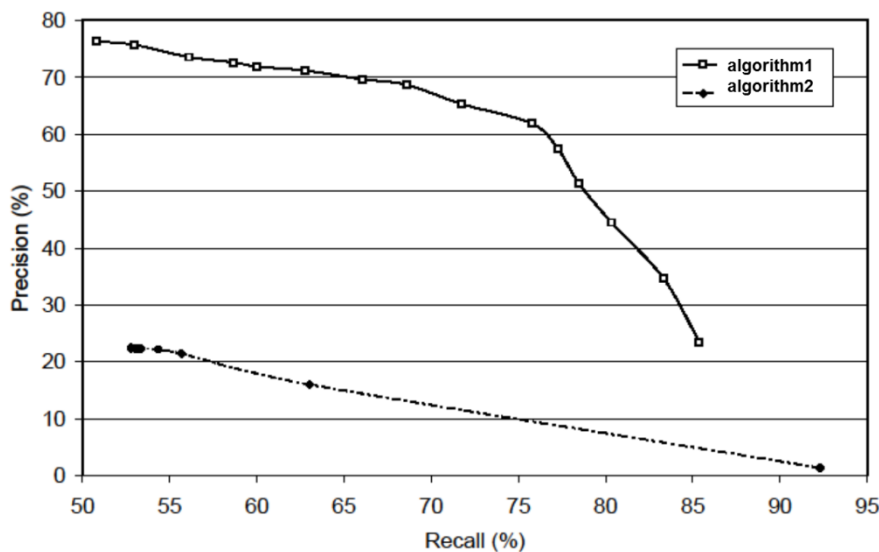
$$\textit{precision} = \frac{\textit{TruePositives}}{\textit{TruePositives} + \textit{FalsePositives}}$$

$$\textit{recall} = \frac{\textit{TruePositives}}{\textit{TruePositives} + \textit{FalseNegatives}}$$

- the higher the precision and the recall, the better

Evaluation (4/4)

- Apply different shot detection techniques, and make use of varying parameter values
 - count correct and wrong detections
 - calculate precision and recall values
 - plot summarizing ROC (Receiver-Operator-Curve)



Outline

- Introduction
- Techniques for video shot detection
 - pixel-wise comparison
 - selective pixel comparison
 - block-based comparison
 - motion estimation
 - histogram-based comparison
- Performance evaluation
- DirectShow
- Assignment

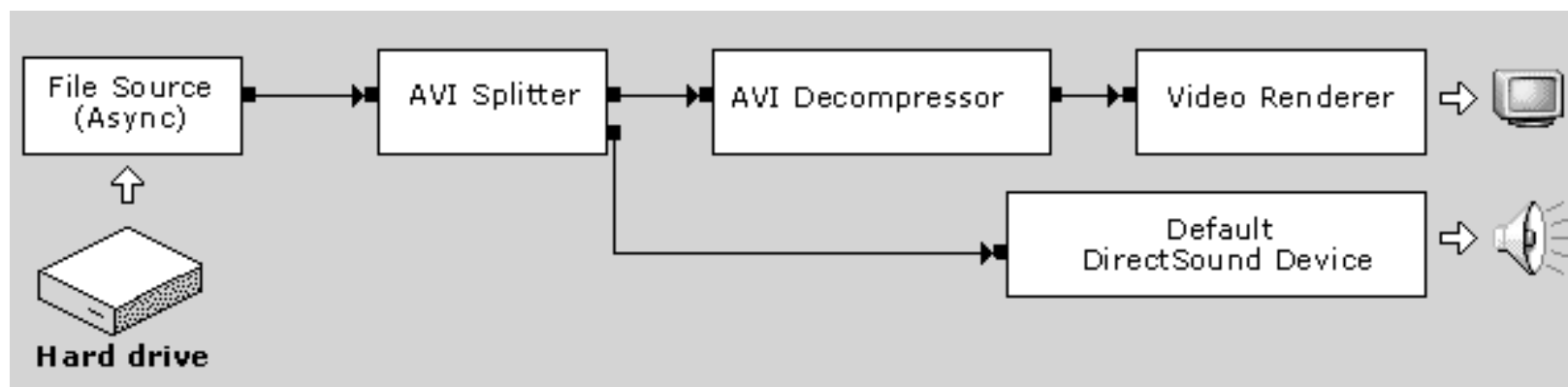
DirectShow (1/3)

- DirectShow
 - framework for processing of multimedia streams
 - successor of the Video for Windows API
 - alternative to the QuickTime API (Apple)
- Designed for C++ & the Component Object Model (COM)
 - complicated but still widely used by the industry
 - in our exercises, a C# wrapper will be used
 - DirectShow.Net Library



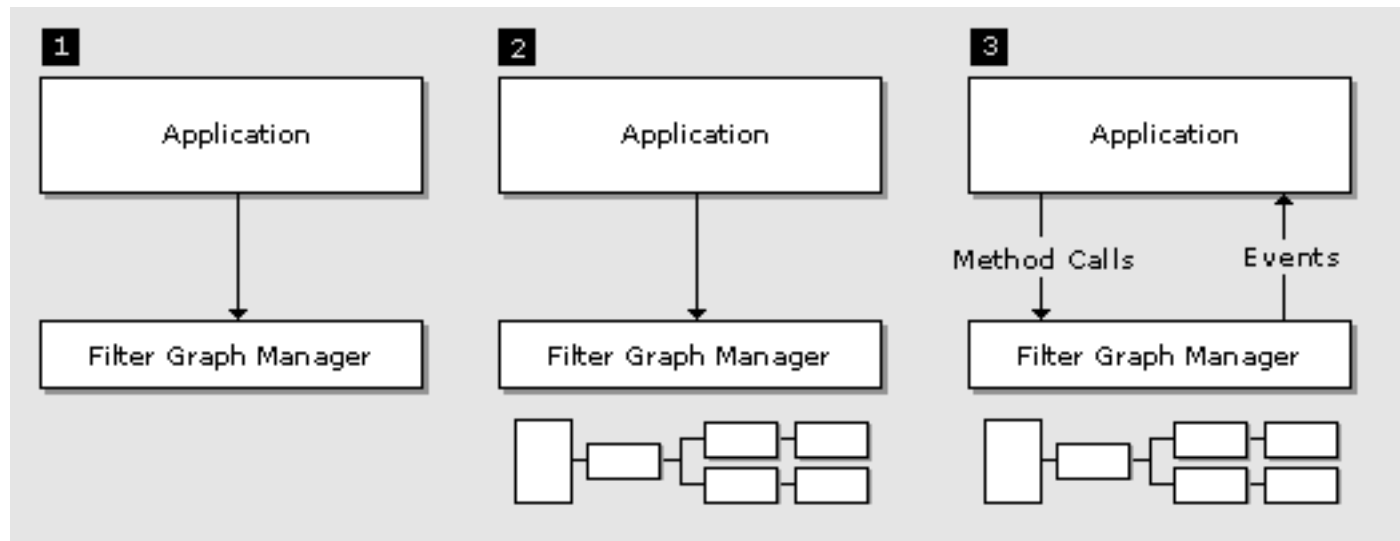
DirectShow (2/3)

- Filters are the building blocks of DirectShow
 - source filter
 - transform filter
 - renderer filter
- Filters are connected into a Filter Graph
 - can be visualized by means of the GraphEdit tool



DirectShow (3/3)

- DirectShow application
 - (1) gets an instance of the Filter Graph Manager
 - (2) builds a filter graph
 - (3) steers the filter graph and responds to events



Outline

- Introduction
- Techniques for video shot detection
 - pixel comparison
 - selective pixel comparison
 - block-based comparison
 - motion estimation
 - histogram-based comparison
- Performance evaluation
- DirectShow
- **Assignment**

Assignment (1/4)

- Create a DirectShow-based C# application
 - needs to come with a GUI
 - required functionality
 - open video
 - play video (see example applications)
 - detect shots (see algorithms)
 - show shots
 - play shot
 - annotate shot
 - export shot info
 - export frames
 - retrieve shots

Assignment (2/4)

- Algorithms

- Exercise 1

- implementation of selective pixel difference

- Exercise 2

- implementation of motion estimation method

- Exercise 3

- implementation of global histogram comparison

- Exercise 4

- implementation of local histogram comparison

- Exercise 5

- implementation of generalized method for shot detection

cuts

cuts &
gradual
transitions

Assignment (3/4)

- Exercise 6: evaluation and report
 - precision and recall, as well as time complexity
 - write a report
 - clear information about the algorithms implemented
 - evaluation of the algorithms implemented
 - answer the questions given
 - important!
 - report has to be clear and well-structured
 - choose the best way to visualize your results
 - size of the report
 - absolute maximum: 6 pages
 - quality above quantity!

Assignment (4/4)

- **Deadlines**
 - software (application + algorithms)
 - December 4th, 16h00 (Thursday)
 - report (in English)
 - December 11th, 16h00 (Thursday)