Image Retrieval through Codebooks

Marián Labuda

Image Retrieval System

- Principles
- System Architecture
- Visual Vocabularies
- Experimental results
- Conclusion

Image Retrieval System

- **Principles**

- local feature extraction from a set of images

- conversion of features to visual words (quantization)

- using text retrieval technology

Marián Labuda

359322@mail.muni.cz

Image Retrieval System

- **Basic operations**

1) Indexing

- Feature extraction - list of local image features
  - features as descriptors (high dimensional vectors)
- Quantization – images as text documents
  - quantization through visual vocabulary
  - simple text string instead of high dimensional vector
- Storage – visual words in inverted index

2) Querying

- Feature extraction – a list of local image features for a given image
- Quantization – the query image as a list of keywords
- Searching – in an inverted index using text retrieval engine
  - answer ranking according to similarity
- Post-processing – e.g. spatial verification

Image Retrieval System

Feature extraction

- image content described by image features

- image features - global features (e.g. color histogram)
  - local features (e.g. SIFT, SURF, MSER, …)

- pros. & cons. of local image features

- extraction of features creates descriptors

- computation complexity of high dimensional descriptors

Marián Labuda
359322@mail.muni.cz

Image Retrieval System

SIFTs

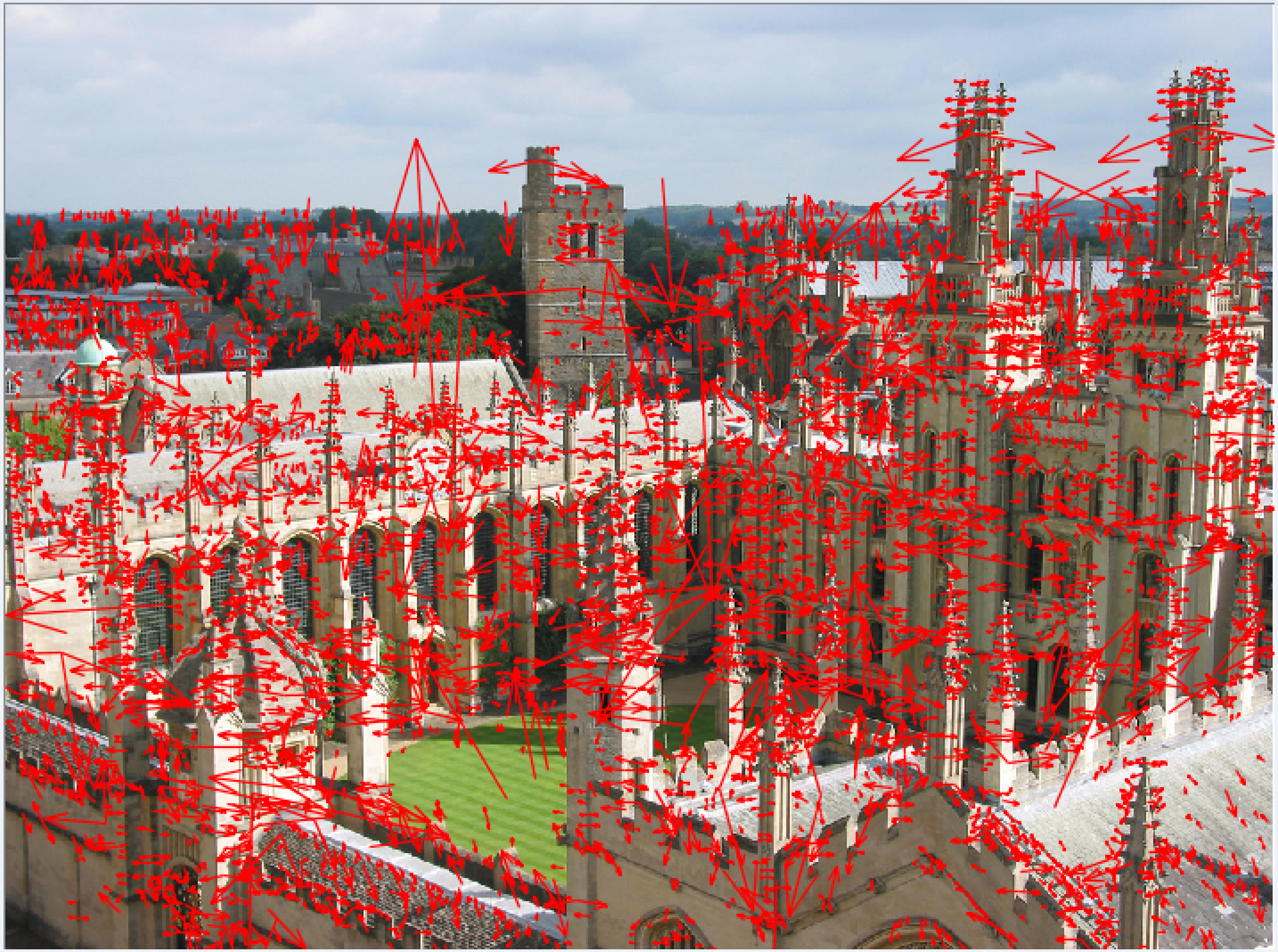


Image Retrieval System

Color Histogram

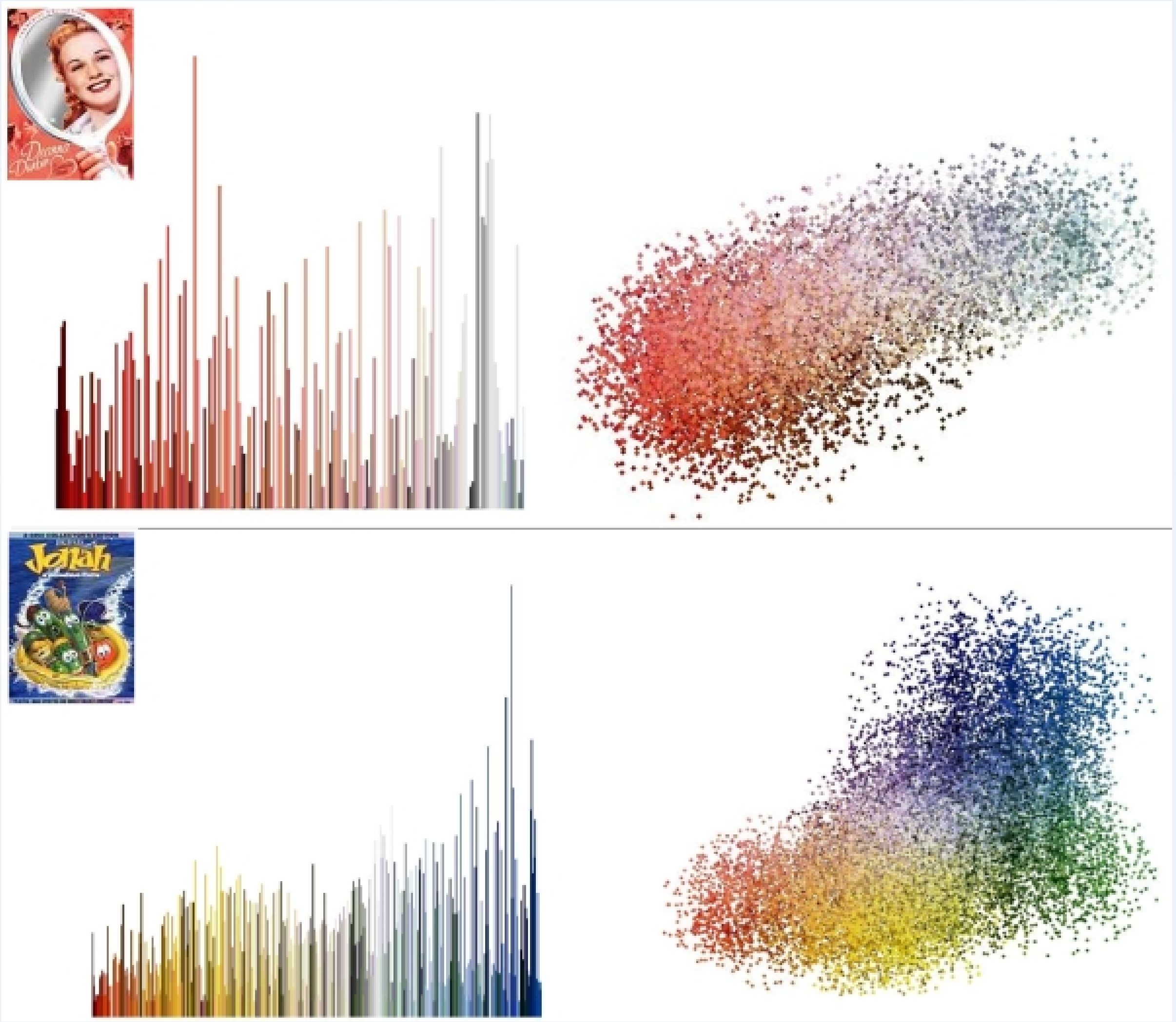


Image Retrieval System

- **System Architecture**

- **MESSIF framework** – data abstraction
  - feature extraction
  - Web UI

- **Lucene Core** – indexing and searching

- **My implementation** - visual vocabulary
  - similarity modification
  - post-processing

Image Retrieval System

- **Lucene Core**

  - full-text search engine

  - usage of inverted files

  - high performance

  - supports various types of queries

  - ranked searching

  - using Lucene is provided through class *LuceneAlgorithm*
    - combines Lucene Core library and MESSIF framework

Marián Labuda

359322@mail.muni.cz

Image Retrieval System

Visual Vocabulary

- provide quantization of local image descriptors

- visual vocabulary as a bottleneck of Image Retrieval System

- images as sets of quantized features

- some words can occur too often, others very few
    - later we can filter them as stop words (analogy to prepositions, ...)

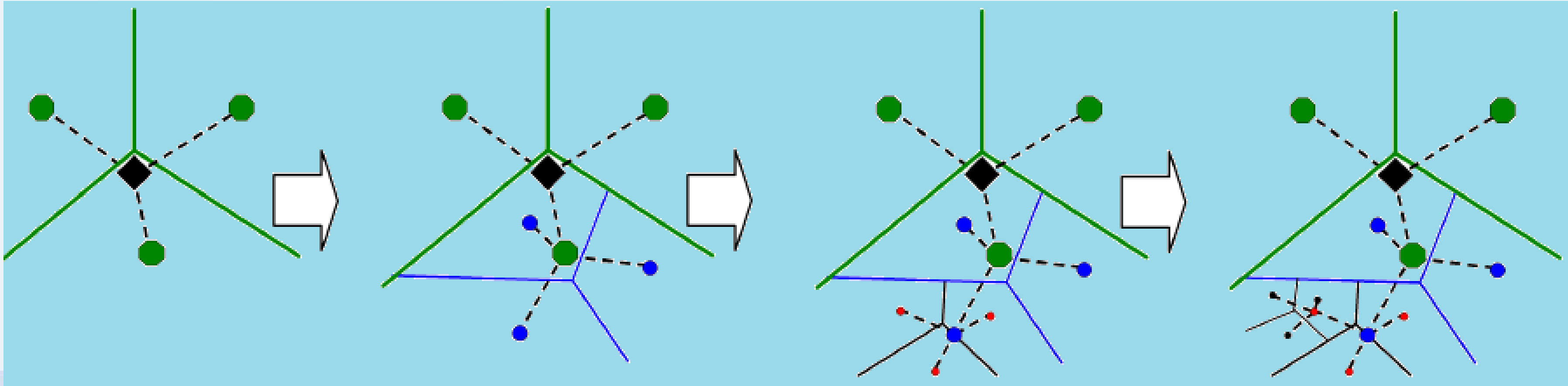


Image Retrieval System

- **My implementation - Visual vocabulary**

 - abstract class *AbstractVisualVocabulary*

 - simple implementation and integration of a new visual vocabulary

 - implemented visual vocabularies:
   - *K-means*
   - *MDPV (metric distance permutation vocabulary)*

- efficiency of the given visual vocabulary

Image Retrieval System

Visual vocabularies - hierarchical k-means

- hierarchical k-means provides a faster way to quantize descriptors than flat

- a visual word as a path from the root to a leaf node

Demo application

- deployed at http://mufin.fi.muni.cz/subimages-lucene/random

- oxford building dataset (~5000 images)

- 6 hierarchical k-means

- 10 descriptors in node

Image Retrieval System

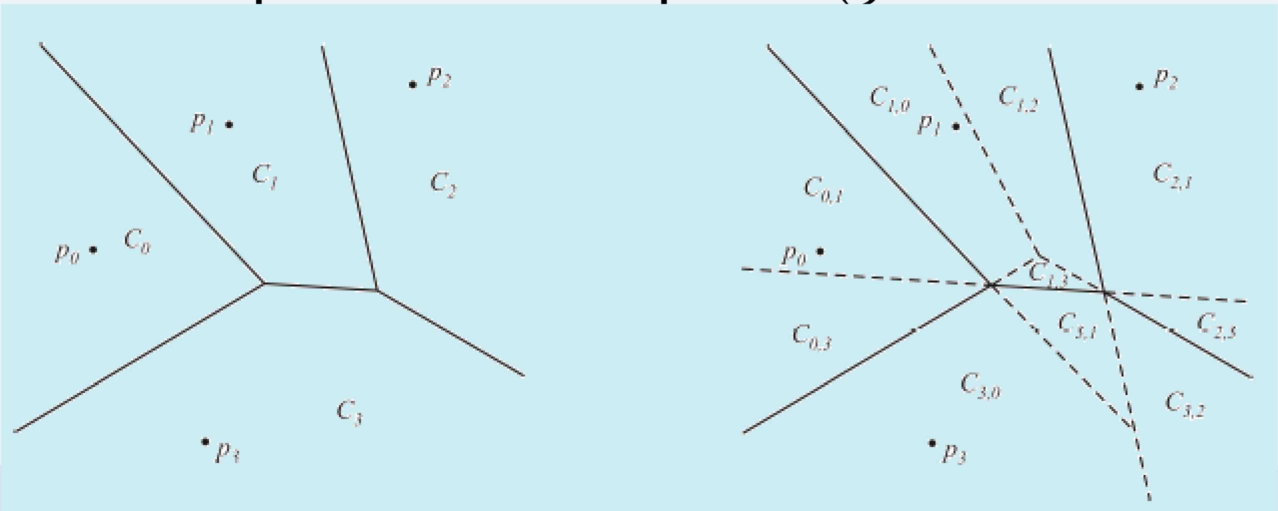
•**Visual vocabularies - MDPV**

- idea of metric distance permutation vocabulary – based on a small number of pivots and recursivelly defines voronoi cells

- provides finer granularity than k-means

- faster and requires less space than k-means

- visual words as a sequence of more pivots (5 or 10 are used at most)



Marián Labuda
359322@mail.muni.cz

Image Retrieval System

- **My implementation - Similarity modification**

  - text based retrieval systems often use tf-idf scoring
    - tf - term frequency – „How many times does the term occur in the document (in one image)?“
    - idf - inverse document frequency – „How many documents (images) contain the given term in whole collection?“

  - Lucene provides an easy way to implement own similarity

  - definition of similarity by scoring

  - score computation:

$$score(q,d) = coord(q,d) * \Sigma(tf(t \in d) * idf(t) * t.getBoost())$$

  - experimentally determined the best similarity on the ukbench dataset

Image Retrieval System

- **My implementation – Post-processing**

- RANSAC - RANdom SAmple Consensus

- geometric verification

- not enough improvement on the given vocabularies – fine-grained words

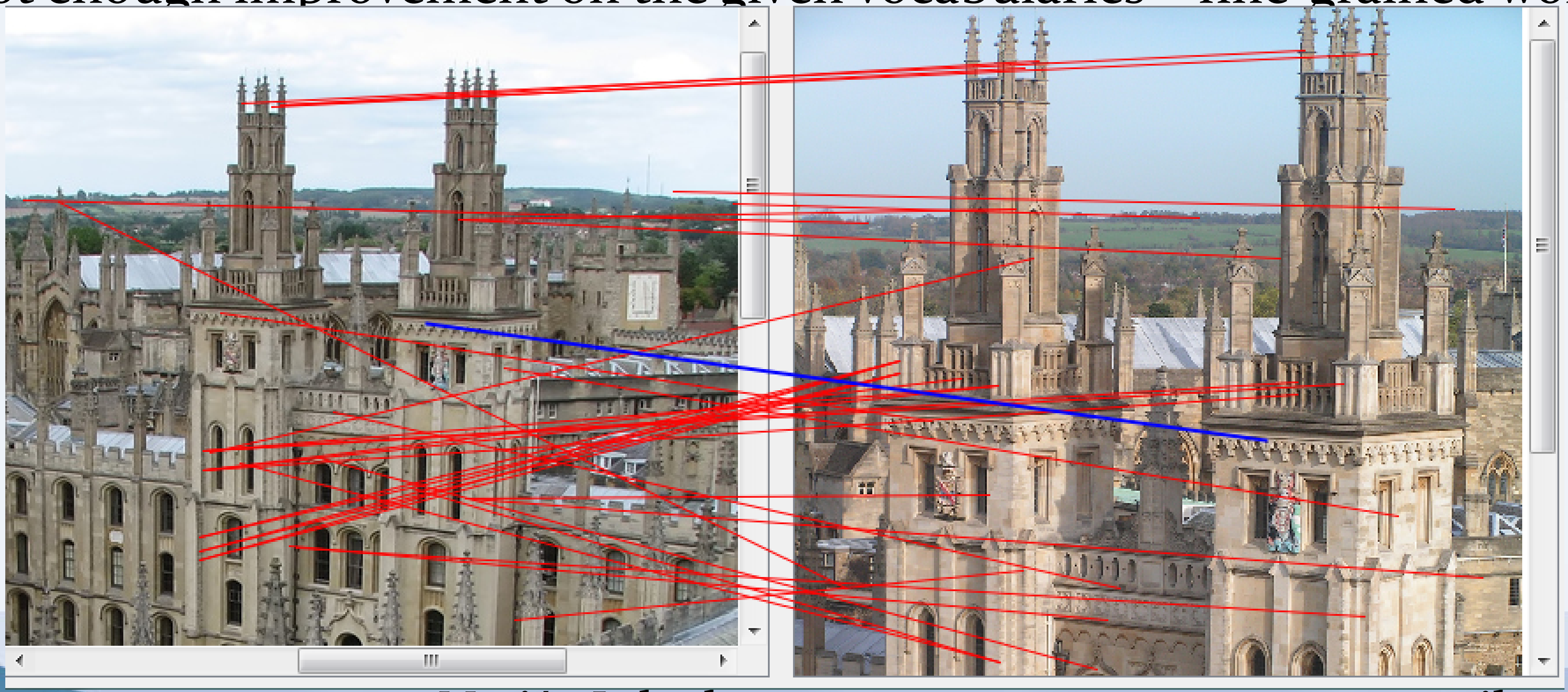


Image Retrieval System

- **Experimental results**

- Ukbench dataset contains 5 (almost) different datasets
  - CD – CD covers
  - moving – moving vehicles
  - CD+moving – combination of CD and moving datasets
  - flip – some flipped versions of normal dataset
  - normal – test set

- locators of similar images example:

  sift100000
  sift100001
  sift100002
  sift100003

Image Retrieval System

- **Experimental results (continue)**

  - similarity tests based on the matrix of different settings:
    - term frequency weights
    - inverse document frequency weights
    - overlap weights

  - compared to ukbench results – visual words provided by authors

  - comparison based on relationship to flat ukbench results
    - slightly better results in effectiveness – better retrieval
    - slightly worse results in speed

  - „best“ similarity: $tf = 1 \quad idf = (\ln \frac{N}{N_i})^2 \quad coord = (\frac{overlap}{maxOverlap})^3$

  N – total count of documents
  overlap – terms of query in document
  N_i - documents count containing term
  maxOverlap – count of query terms

Image Retrieval System

- **Experimental results (continue 2)**

- how many of the ground-truth images occured in top 4:

Dataset	My Similarity	Customized	Ukbench
CD	2,9583	2,9344	2,8956
movie	2,8912	2,8762	2,8285
CD & movie	2,9528	2,9326	2,8844
flip	2,9609	2,9548	3,0144
test	3,1414	3,1498	3,1664
Average value	2,9809	2,9696	2,9579

- each dataset consists of 10200 images
- every image has 3 other ground-truth images
- Customized similarity is same as My Similarity except the tf
- customized similarity count with term frequency as $\sqrt{f}$ instead of 1

Image Retrieval System

- **Experimental results (continue 3)**

 - unfortunately I was not able to reproduce their results with any similarity

 - stop words did not provide more precise results

 - term frequency can be fully omited

 - index time is up to 2 minutes

 - searching of 10200 queries on database consisting of 10200 documents takes from 6 to 10 minutes (39 – 60 ms per image)

 - experiments on Lenovo T430s – 8 GB RAM
 - Intel Core i7 3520M (2,9 GHz, 2 cores)

Image Retrieval System

- **Conclusion**

  - image retrieval system based on visual vocabularies and inverted files are as strong as the vocabulary

  - term frequency can be fully omited

  - usage of stop words depends on vocabulary

  - inverted index provides fast and scalable solution