Multimedia Retrieval Exercise Course

4 Query by Example: Color Histogram Extraction

Kimiaki Shirahama, D.E.

Research Group for Pattern Recognition
Institute for Vision and Graphics
University of Siegen, Germany

Overview of Today’s Lesson

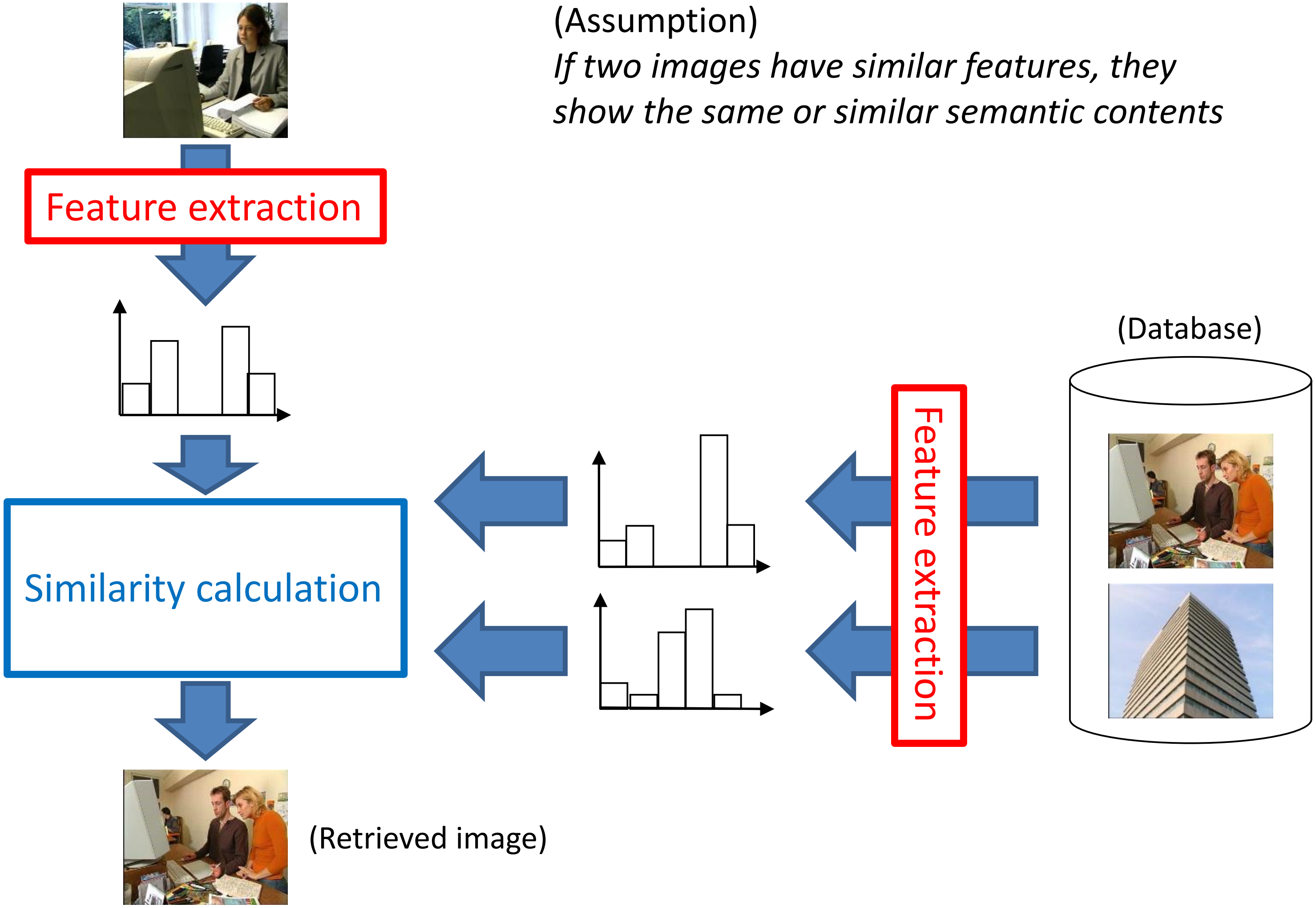
1. Query by Example (Content-Based Image Retrieval (CBIR))
2. Color Histogram Extraction by OpenCV
3. Extracting Histograms from All Images

Query by Example
(Content-based Image Retrieval)

Given sample images as a query, retrieve similar images from the database

(Assumption)
If two images have similar features, they show the same or similar semantic contents

(Query: People appear with computers)



Feature extraction

Similarity calculation

Feature extraction

(Database)

(Retrieved image)

Implementing Query by Example

Pre-processing phase: Extract features from all images in the database, and store them into a file (In this course, we will use histogram-type features)

```
File f;
for each image I in the database
   histo = extractHistogram(I);
   outputHistogram(f, histo);
end
```

Retrieval phase: Compute similarities between a query image and the other images in the database, and output images with the highest similarities.

```
Query q;
q_histo = loadHistogram(q);
Result r;
For each image I in the databse
  histo = loadHistogram(I);
  sim = computeSimilarity(q_histo, histo);
  addResult(r, I, sim);
end
sortBySimilarity(r);
outputSimilarImages(r);
(Evaluation of the retrieval result)
```

Color Histogram Extraction by OpenCV (1/2)

```
void cv::calcHist(const Mat* images, int nimages, const int* channels, InputArray mask,
                  OutputArray hist, int dims, const int* histSize, const float** ranges,
                  bool uniform=true, bool accumulate=false)
```

- images: Input image(s)
- nimages: The number of image(s) from which a histogram is extracted
- channels: The channel IDs for which a histogram is extracted
- mask: Mask to focus on a certain region in an image
- hist: Histogram to be extracted (cv::Mat)
- dims: Histogram dimensionality
- histSize: The number of bins in each histogram dimension
- ranges: The value range for each histogram dimension
- uniform: Uniform arrangement of bin boundaries (true) or non-uniform one (false)
- accumulate: Accumulate histogram values (true) or not (false)

For more detail, please refer to
http://docs.opencv.org/3.1.0/d6/dc7/group__imgproc__hist.html#ga4b2b5fd75503ff9e6844cc4dcdaed35d
A sample code for histogram extraction is also provided.

Color Histogram Extraction by OpenCV (2/2)

We will create a histogram from an image in the HSV color space

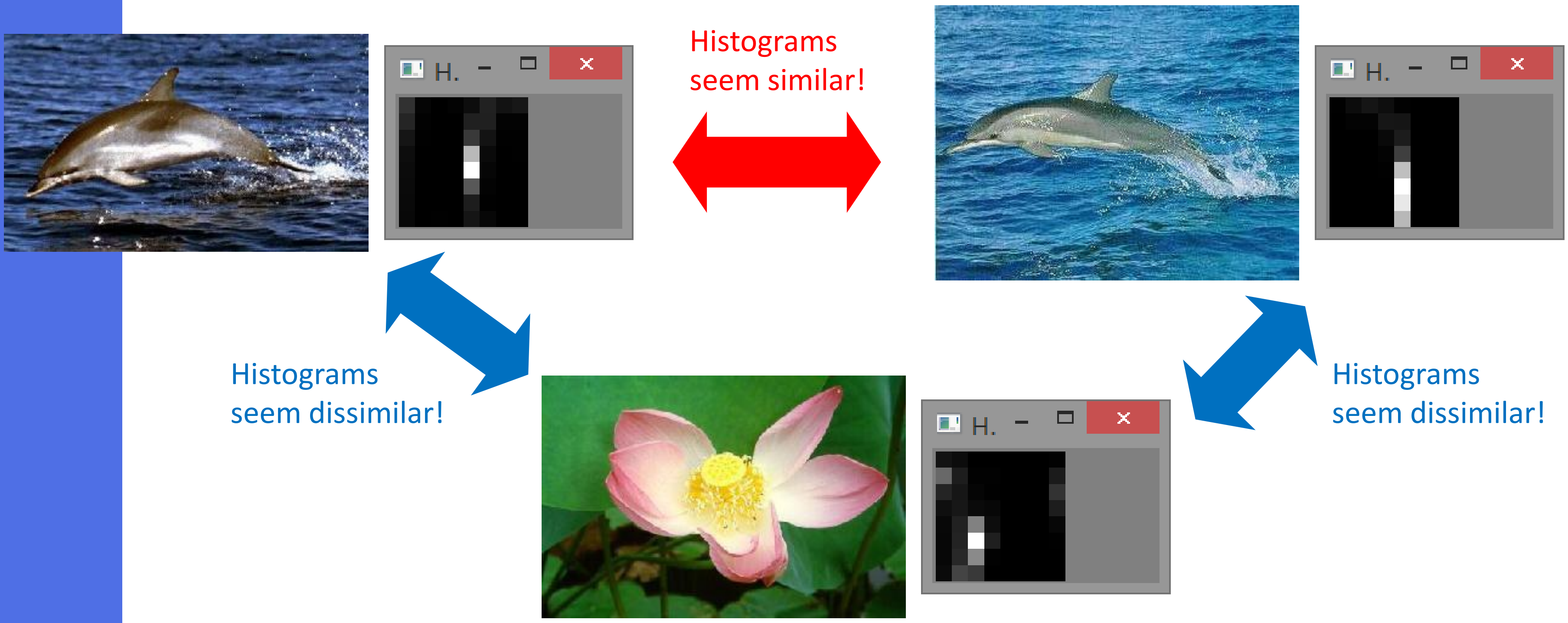
```
// Parameters for creating a histogram
int hist_dims = 3; // 3-dimensional histogram (each dimension corresponds to H, S or V axis)
int h_bins = 8;
int s_bins = 8;
int v_bins = 8;
int hist_size[] = { h_bins, s_bins, v_bins }; // # of bins on H, S and V axes
float h_ranges[] = { 0, 180 }; // Values on H axis range from 0 to 180
float s_ranges[] = { 0, 256 }; // Values on S axis range from 0 to 255
float v_ranges[] = { 0, 256 }; // Values on V axis range from 0 to 255
const float *ranges[] = { h_ranges, s_ranges, v_ranges };
int channels[] = { 0, 1, 2 };

cv::Mat hist;
calcHist(&hsv, 1, channels, cv::Mat(), hist, 3, hist_sizes, ranges);
```

FYI: The memory region used for cv::Mat is automatically released when it is not used any more.

Visualising Histograms

Please try to visualise an extracted histogram by referring to the Web page.
Note: The example code in the Web page targets a two-dimensional histogram. On the other hand, we target a three-dimensional histogram. To reduce it into a two-dimensional image, you need to take a sum of bin values by fixing two of three dimensions.



Histograms seem similar!

Histograms seem dissimilar!

Histograms seem dissimilar!

Every time you write a code, I strongly recommend you to check if your program works or not (using simple data)!

Extracting Histograms of All Images in Caltech 101 dataset

Please try to extract histograms from all the 9144 images in Caltech 101 (should be downloaded based on last lesson's slides), and store them in a single text file

1. It's better to separate the histogram extraction process from the main function.

2. After you finish your implementation in the "debug" mode, it's better to run the code in the "release" mode
To use this mode, you need to specify the include and library directories, and specify "opecv_world310.lib".

3. How to list directory or file names?

- Windows: FindFirstFile, FindNextFile
  (In my case, I had to set the character code to "multi-bytes" in order to use Japanese filename)
- Linux: opendir, popen, etc.
- Mac: I don't know ☺ But, I think the same way as Linux can be used, because, to my knowledge, Mac is based on debian-based linux.

4. File writing is clear (use fopen or ofsteam)

5. Writing format is up to you. I made the file where each line is as follows:
<Image filename> <Bin value at (0,0,0) position> <Bin value at (0,0,1) position> … <Bin value at (7,7,7) position>

NOTE: Once you have made a text file, you don't have to use OpenCV any more. This is because you only have to read the text file, and compute similarities among histograms.