Multimedia Retrieval Exercise Course

3 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. Image Data Preparation (Caltech 101)
4. Extracting Histograms from All Images

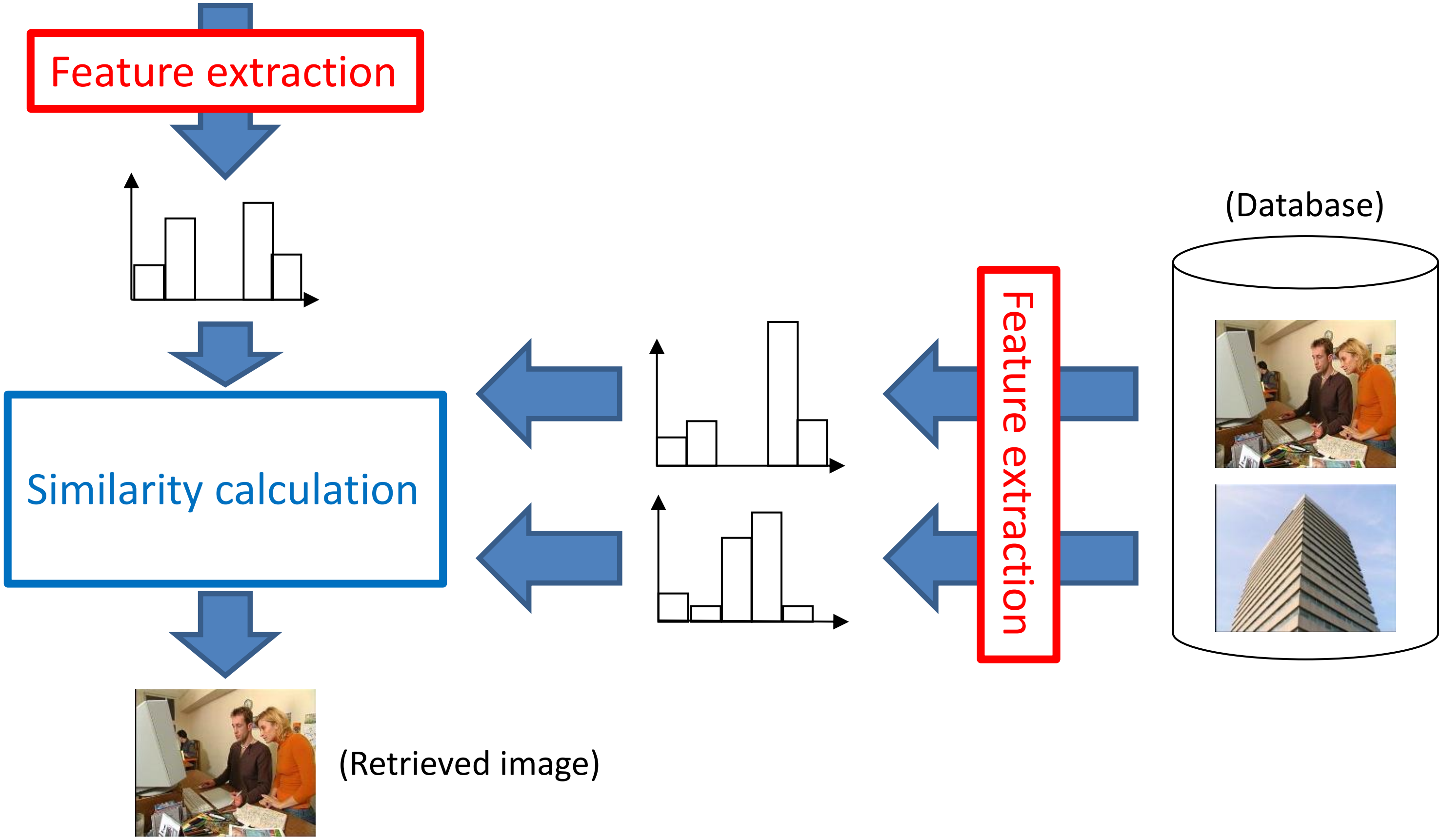
Query by Example
(Content-based Image Retrieval)

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

(Query: People appear with computers)

(Assumption)

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



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
```

This week

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)
```

The next week and the week after that

Color Histogram Extraction by OpenCV (1/3)

1. CvHistogram* cvCreateHist(int dims, int* sizes, int type, float** ranges=NULL, int uniform=1)

Allocate the memory region of a histogram

- dims: Number of dimensions
- sizes: Integer array where each element represents the number of bins in one dimension
- ranges: Two-dimensional array where each element represents the upper or lower bound of values in one dimension

For other input variables, please refer to http://opencv.jp/opencv-1.1.0_org/docs/ref/opencvref_cv.htm#cv_imgproc_histograms

```
typedef struct CvHistogram {
        int type;
        CvArr* bins;
        float thresh[CV_MAX_DIM][2];
        float** thresh2;
        CvMatND mat;
 }
CvHistogram;
```

Color Histogram Extraction by OpenCV (2/3)

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, 181 }; // 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
float *ranges[] = { h_ranges, s_ranges, v_ranges };

CvHistogram* hist = cvCreateHist(hist_dims, hist_size, CV_HIST_ARRAY, ranges, 1);
```

Color Histogram Extraction by OpenCV (3/3)

2. Split a target HSV image into three images, each represents pixel values on a single channel
void cvSplit(const CvArr* src, CvArr* dst0, CvArr* dst1, CvArr* dst2, CvArr* dst3)

- src: Source image represented by multiple color channels
- dst0: An image representing pixel values in the first channel (In our case, H channel)
- dst1: An image representing pixel values in the second channel (In our case, S channel)
- dst2: An image representing pixel values in the second channel (In our case, V channel)
- dst3: An image representing pixel values in the second channel (In our case, NULL)

```
IplImage *planes[3];
planes[0] = cvCreateImage(cvSize(src->width,src->height), src->depth, 1);
planes[1] = cvCreateImage(cvSize(src->width,src->height), src->depth, 1);
planes[2] = cvCreateImage(cvSize(src->width,src->height), src->depth, 1);
// Split the HSV image into three images, each of which represents pixel values on a signle channel
cvSplit(hsv, planes[0], planes[1], planes[2], NULL);
```

3. Extract a histogram with the following function
void cvCalcHist(IplImage** image, CvHistogram* hist, int accumulate=0, const CvArr* mask=NULL)

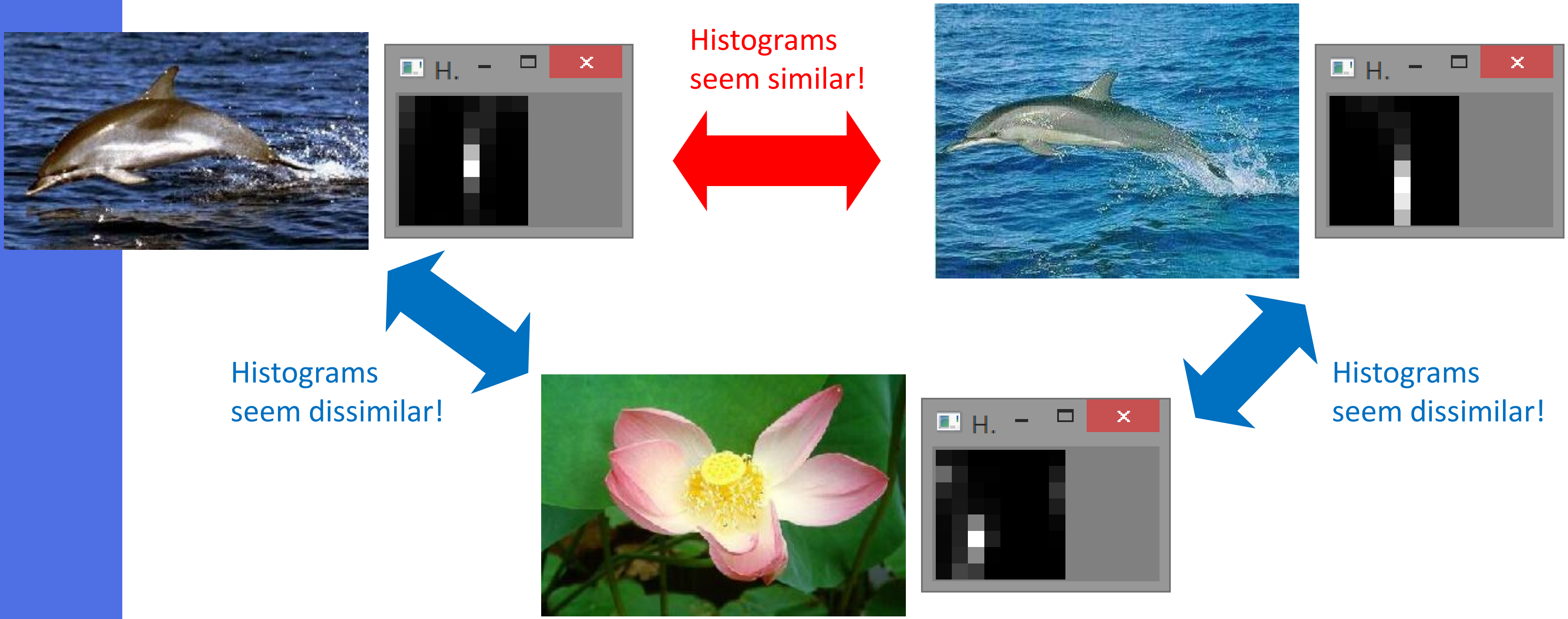
- image: Array of IplImage pointers for split images
- hist: Pointer to the CvHistogram

For the other variables, please refer to the Web page

NOTE: Don’t forget releasing the memory region of the histogram
by cvReleaseHist(CvHistogram** hist), after the task for it is finished.

Visualising Histograms

Please try to visualise the 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!

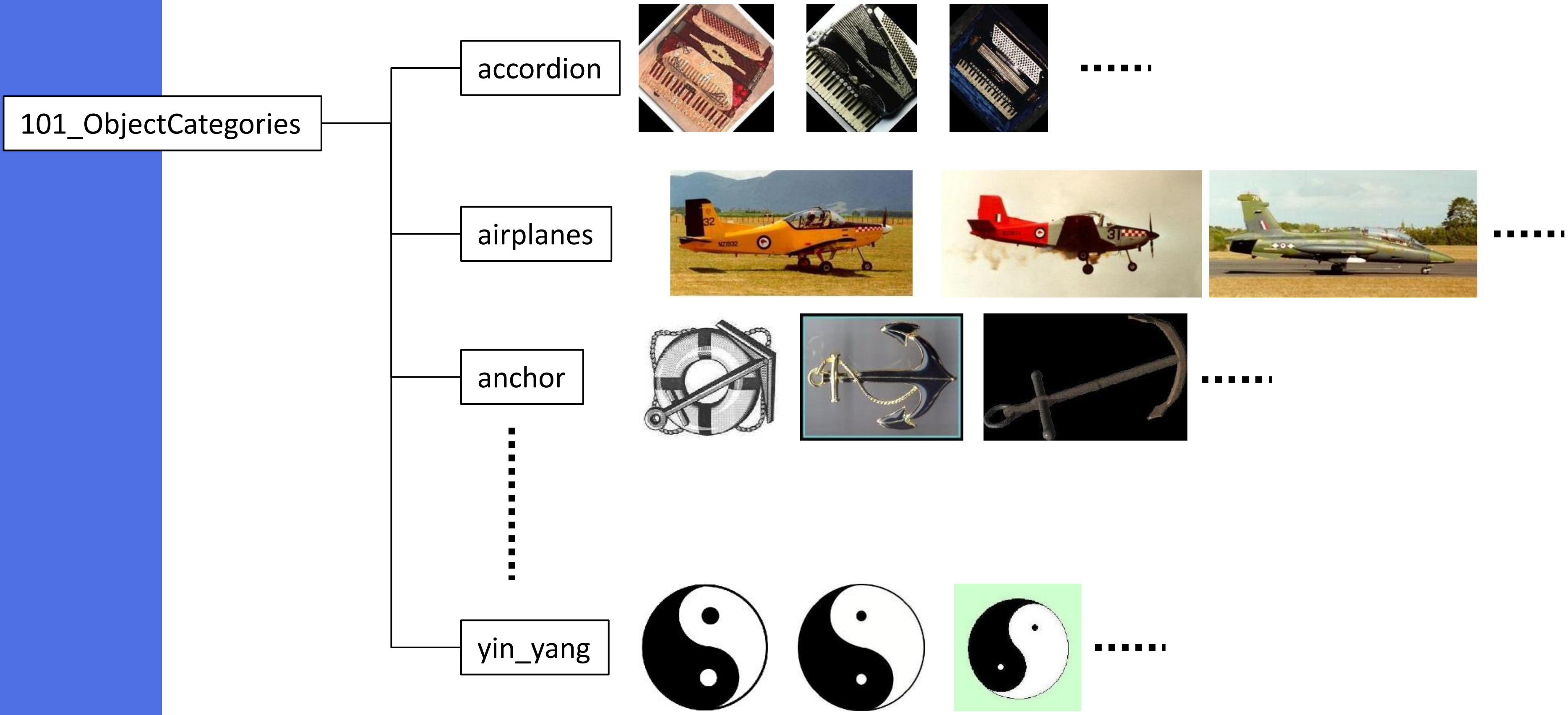
Histograms seem dissimilar!

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

Image Data Preparation

Caltech 101 (http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html)

- Traditional benchmark image data set for evaluating image retrieval/classification methods
- This includes 101 categories, each category contains about 40 to 800 images
- Total data size is about 150MB



101_ObjectCategories

accordion

airplanes

anchor

yin_yang

Extracting Histograms of All Images

Please try to extract histograms from all images, and store them in a single file

Problems that you have to solve:

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

2. How to list directory or file names?

- Windows: FindFirstFile, FindNextFile
- Linux: opendir, popen, etc.
- Mac: I don’t know ☺ But, I think the same way to Linux can be used, because, to my knowledge, Mac is based on debian-based linux.

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

4. 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: In the next lesson, we will not use OpenCV. The reason is that we only have to read the text file of histograms, and compute their similarities.