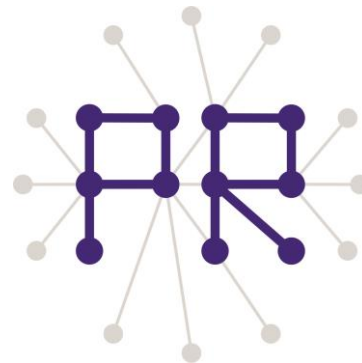# Multimedia Retrieval Exercise Course

7 Local Features (SIFT: Scale-Invariant Feature Transform)

Kimiaki Shirahama, D.E.

Research Group for Pattern Recognition
Institute for Vision and Graphics
University of Siegen, Germany
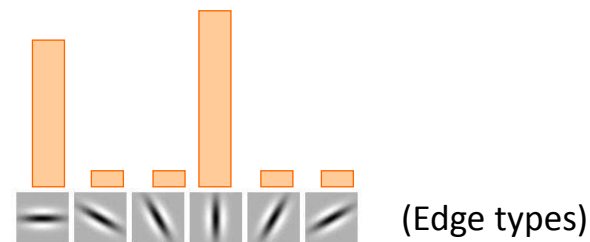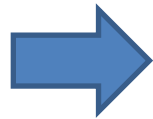
Overview of Today's Lesson

- Problem of global features
- Local features
  - ❑ Extraction of local features
- SIFT feature
  - ❑ Difference-of-Gaussian (DoG)
  - ❑ SIFT descriptor
- SURF feature
- Extraction of SURF Features by OpenCV

Problem of Global Features

Global features: Features extracted from the whole region of an image

(Color types)

(Edge types)

(Trajectory directions)
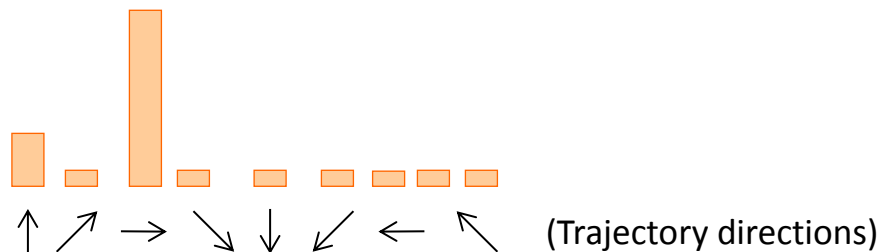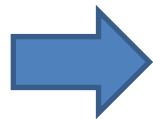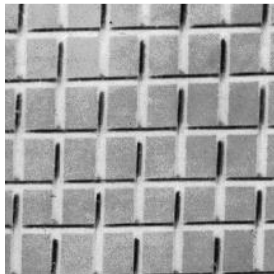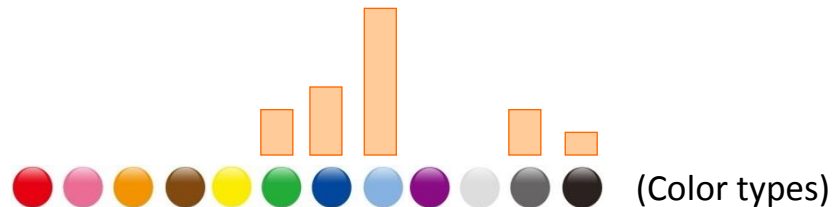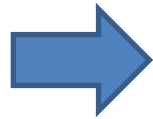
Too coarse to deal with the detailed information in images
(We already experienced the semantic gap when using color histograms)

Local Features

Local features: Features extracted from a local region of an image

Local features enable much more detailed analysis than global features
Examples of local features: SIFT, HOG, ColorSIFT, SURF, MSER, etc.

Extraction of Local Features

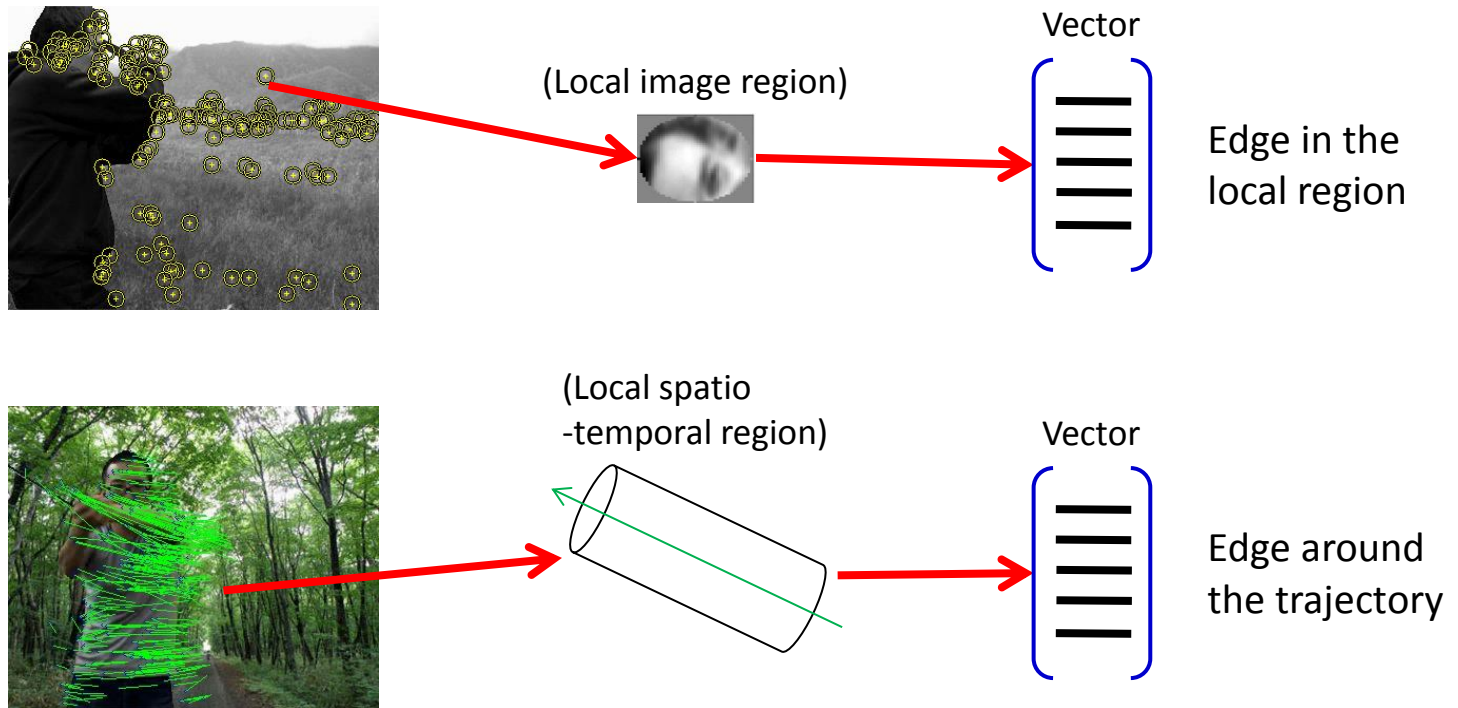Local features are extracted based on the following two steps
1. Region detector: Determine local regions from which features are extracted
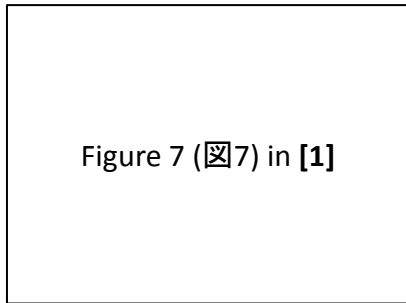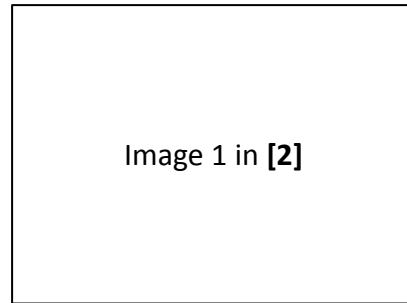
Figure 7 (図7) in **[1]**	Image 1 in **[2]**	
(Difference-of-Gaussian)	(Harris-Affine detector)	(Dense sampling)
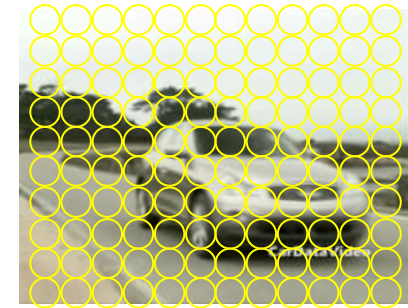
Determine yellow regions in the above images

2. Region descriptor: Describe each local region with a vector representation
- Edge distribution
- Color distribution
- Edge distribution on each color channel

etc.

SIFT Feature

Scale-Invariant Feature Transform (SIFT): Represents the edge shape in a local region, reasonably invariant to scaling, rotation, viewpoint changes and illumination changes

➡ The most popular feature for multimedia retrieval

(SIFT feature has originally developed to *match points of a 3D object in different images*)
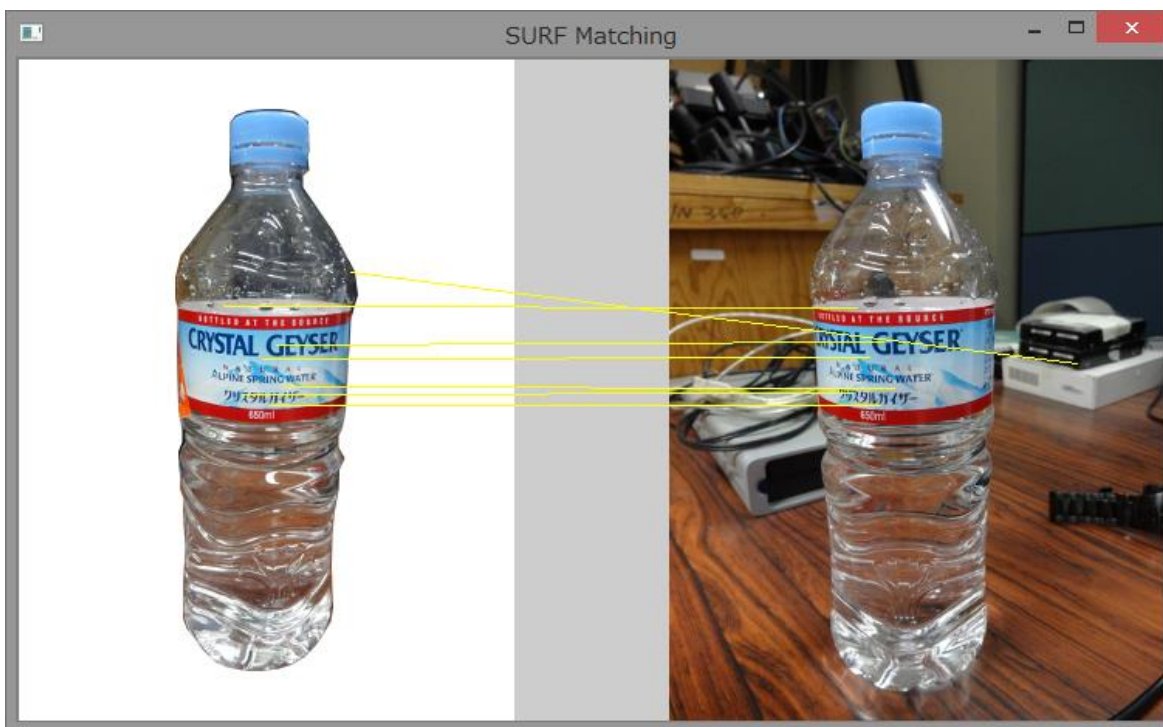
Specific object recognition: Identify the same object instance in different images
(**Generic object recognition:** Identify the class of an object in an image)
Explain the most basic SIFT feature using DoG as a region detector

Difference-of-Gaussian (DoG) (1/2)

What kind of points (regions) are useful for matching?

➡️ *Regions where pixel values largely change in multiple directions*

Detect local regions based on the difference of images, which are smoothed by Gaussian filters with different scales

➡️ Regions where the difference is large contain many edges!

Figure 2 (図2) in **[1]**

(Gaussian Filter (1-dimensional))
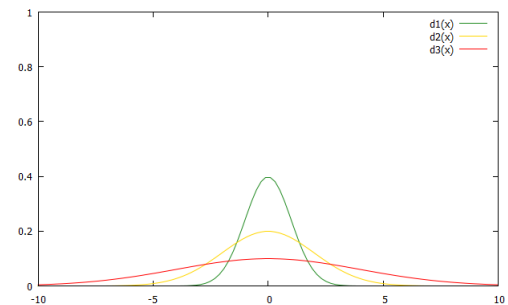
Difference-of-Gaussian (DoG) (2/2)

σ_3

σ_2

σ_1

Figure 5 (図5) in **[1]**
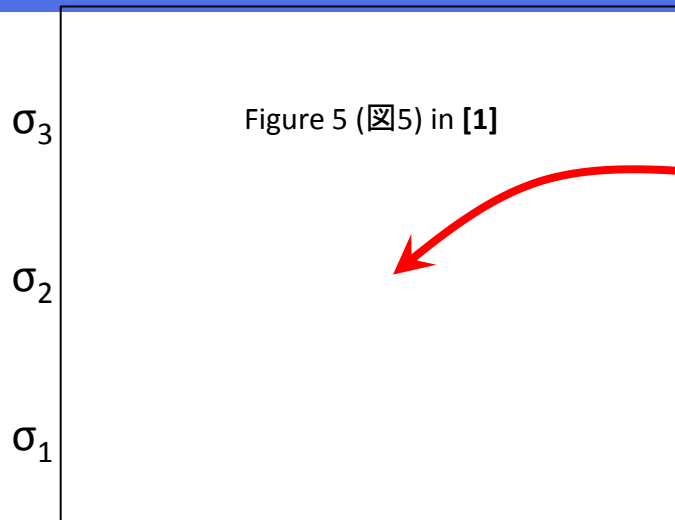
Check whether the DoG of this pixel is extremum or not (the DoG is larger than those of surrounding pixels)

If the DoG is an extremum, the region which surround this pixel with the scale σ_2, is detected as a local region.

Figure 6 (図6) in **[1]**

If the image size becomes two-times larger, the region with the two-times larger σ becomes a local region

Scale invariant!

SIFT Descriptor

Right image in Figure 10
(図10) in **[1]**

Figure 11 (図11) in **[1]**
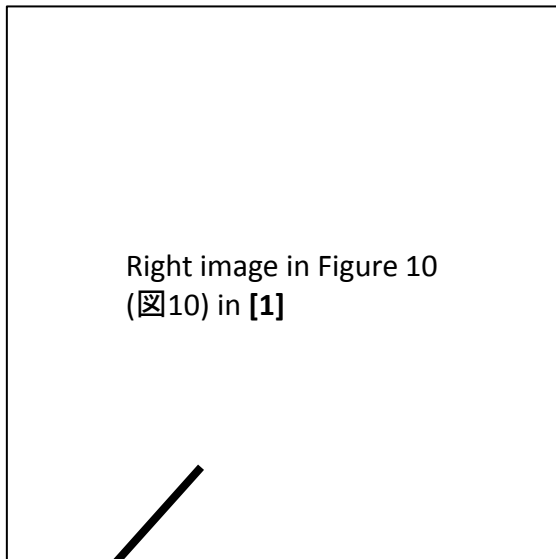
1. Compute the orientation of a local region

2. Rotate the local region so that its orientation becomes upward

Rotation invariant!

3. Divide the local region into 4 x 4 blocks and create a 128-dimensional histogram representing the distribution of orientations in each block

Normalized by the total of orientations in the local region *Robust to illumination change!*

(Basic idea of orientation computation)

(u, v)

Equations 22-24 in **[1]**
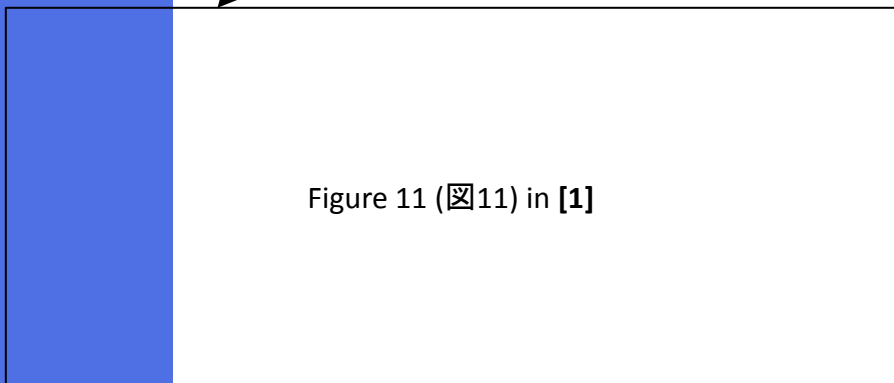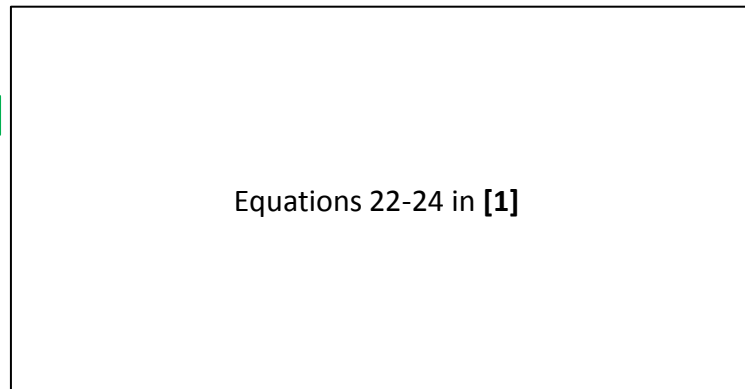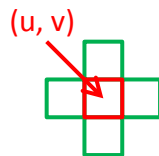
Further processing is done in the actual SIFT descriptor computation

Examples of SIFT Descriptors

(Original image)

(Rotation)

(Scaling)

(Illumination change)

Figure 12 (図12) in **[1]**
- Figure 12 (a): Original image
- Figure 12 (b): Rotated image
- Figure 12 (c): Scaled image
- Figure 12 (d): Dark image
- Figure 12 (f): Image by affine transformation

(Viewpoint change)

For various factors, extracted SIFT descriptors are similar!

This property is due to the invariance to scaling and rotation, but this is not theoretically supported.

SURF Features

Speeded-Up Robust Feature (SURF)

Simplified SIFT feature using the integral image structure

➡️ Compared to SIFT feature, the computation of SURF feature is much more efficient, the performance is slightly worse,

Integral image: Structure where computing the sum of pixel values in any region can be done in O(1) (The most famous application is real-time face detection)

$$76 - 20 - 20 + 9 = 45$$

If the integral image has been created, the sum can be computed in O(1)!

(Original image)
To compute the sum of pixel values in the red region, a double-for-loop has to be used.

(Integral image)
For each pixel, the value represents the sum of pixel values in the rectangle, whose top-left and bottom-right is the origin and this pixel.

```
#include "opencv2\opencv.hpp"
#include "opencv2\nonfree\nonfree.hpp"

#pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_nonfree246d.lib")

#pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_nonfree246.lib")

int main(int argc, char* argv[]){
    cv::initModule_nonfree(); // Very Important: Initialization of the nonfree library

    // Load an image in gray-scale mode (Used to SURF feature extraction)
    // Load the same image in color mode (Used to display extracted SURF features)

    CvMemStorage* storage = cvCreateMemStorage(0);
    CvSeq* keypoints = 0;
    CvSeq* descriptors = 0;
    CvSURFParams params = cvSURFParams(500, 1);

    // Extract SURF features from img
    cvExtractSURF(img, 0, &keypoints, &descriptors, storage, params);
    cout << ">> # of extracted SURF features = " << descriptors->total << endl;

    // Draw extracted keypoints (local regions from each a SURF descriptor is extracted)
    for(int i = 0; i < keypoints->total; i++){
        CvSURFPoint* point = (CvSURFPoint*)cvGetSeqElem(keypoints, i);
        CvPoint center; // Center of a keypoint
        int radius; // Radius of a keypoint (local region size)
        center.x = cvRound(point->pt.x);
        center.y = cvRound(point->pt.y);
        radius = cvRound(point->size * 1.2 / 9.0 * 2.0);
        cvCircle(img2, center, radius, cvScalar(0,255,255), 1, 8, 0);
    }

    // Show the drawn image using cvNamedWindow

    cvClearSeq(descriptors);
    cvClearSeq(keypoints);
    cvReleaseMemStorage(&storage);
    // Other variables should be released here
}
```
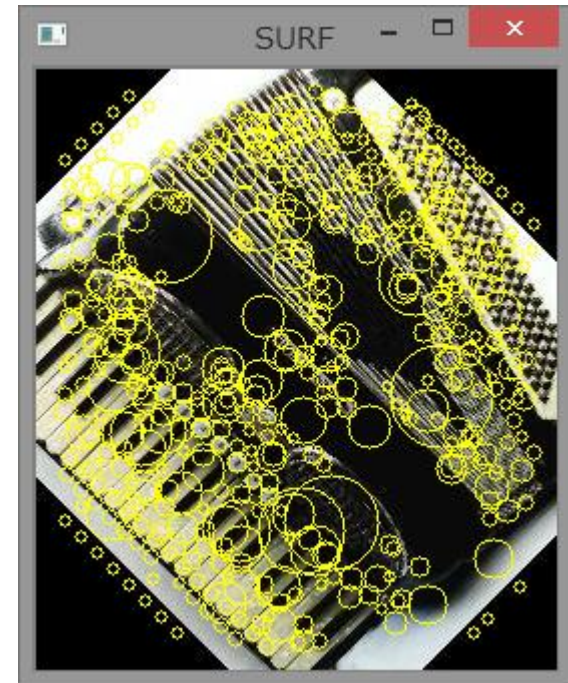
Extraction of SURF Features by OpenCV (1/2)

void cvExtractSURF(const CvArr* image, const CvArr* mask, CvSeq** keypoints, CvSeq** descriptors,
                           CvMemStorage* storage, CvSURFParams params)

- image: 8bit, gray-scale image
- mask: Mask (Not used in this course)
- keypoints: double pointer to the sequence of keypoints (local regions) each is stored in **CvSURFPoint**
    typedef struct CvSURFPoint{
        CvPoint2D32f pt; // keypoint position
        int laplacian;   // -1, 0 or +1 (If two keypoints have different laplacians, they should not be matched)
        int size;   // region size
        float dir;   // orientation
        float hessian;   // used to roughly estimate the strength of keypoint
    } CvSURFPoint;

- descriptors: double pointer to SURF descriptors (each descriptor is 64 or 128-dimensinoal vector of CV_32F
                         (We will use this in the next lesson)
- storage: Storage for storing the actual data of keypoints and descriptors
- params: Parameters for SURF extraction, specified by **CvSURFParams**
    typedef struct CvSURFParams{
        int extended;   // 0: Basic SURF (64-dimensional), 1: Extended SURF (128-dimensional)
        double hessianThreshold; // Threshold used to detect keypoints (local regions)
        int nOctaves;   // # of Gaussian filters used for keypoint detection (default 3)
        int nOctaveLayers;   // # of layers in each octave (default 4)
    } CvSURFParams;

References

[1] H. Fujiyoshi: "Gradient-Based Feature Extraction - SIFT and HOG –", CVIM 160, pp. 211-224 (2007)
http://www.vision.cs.chubu.ac.jp/sift/PDF/sift_tutorial.pdf
[2] Affine Covariant Features
http://www.robots.ox.ac.uk/~vgg/research/affine/index.html

Useful software for SIFT extraction

[3] Koen van de Sande: "ColorDescriptor Software"
http://koen.me/research/colordescriptors/