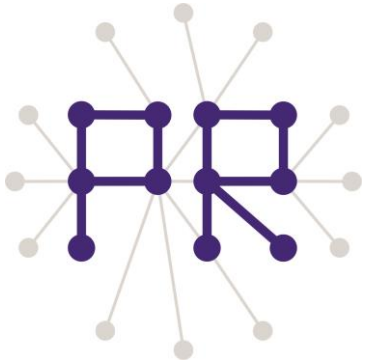
Intro to Multimedia Retrieval Exercise Course

1 Introduction and OpenCV installation

Kimiaki Shirahama, D.E.

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



About This Course

Place/Time

- H-B 8409/10
- 12:15-13:45

Lecturer

- Dr. Eng. Kimiaki Shirahama
- kimiaki.shirahama@uni-siegen.de

Recommendation

- Students in the Bachelor course

Main Purpose

- Study/Exercise basics of multimedia data processing and retrieval

Schedule of the Course

1. **22.10** Introduction and OpenCV installation
2. **29.10** Running a simple program of OpenCV
3. **29.10** Basic image processing by OpenCV
4. **12.11** Query by example (1/3): Color histogram
5. **19.11** Query by example (2/3): Similarity computation
6. **26.11** Query by example (3/3): Finishing the system
7. **03.12** Specific object recognition (1/4): Local features (SIFT and SURF)
8. **10.12** Specific object recognition (2/4): Matching local features
9. **17.12** Specific object recognition (3/4): Finishing the system
10. **24.12** Specific object recognition (4/4): Fast retrieval (KD-Tree and LSH)
11. **07.01** Generic object recognition (1/4): Bag of Visual Words (BoVW)
12. **14.01** Generic object recognition (2/4): Creating BoVW representation
13. **21.01** Generic object recognition (3/4): Classification (libSVM)
14. **28.01** Generic object recognition (4/4): Finish the system
15. **04.02** Discussion

One day before each lesson, the slides will be uploaded the following Web site:
http://www.pr.informatik.uni-siegen.de/en/multimedia-retrieval-1

Requirements for This Course

1. Take your own laptops

2. Programming skill for C is necessary for implementing the codes
   (C++ is desirable, but is not necessary)

3. In the oral examination, you will be asked several questions about this course

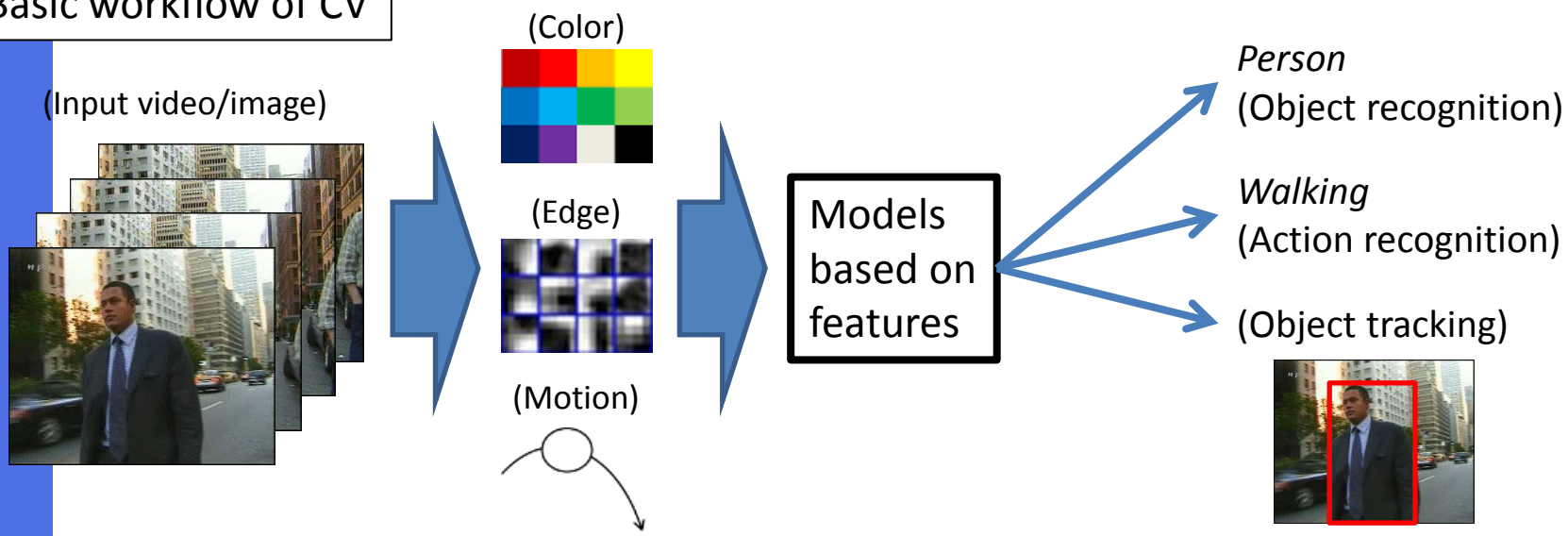
What is OpenCV?

Open-source library consisting of programming functions for ***Computer Vision (CV)***

Implement functionalities of human eyes on computers

- Object recognition
- Object tracking
- Scene understanding
- 3D model reconstruction …. etc.

Basic workflow of CV



Many functions in this workflow have been implemented in OpenCV!

OpenCV Installation (Windows)

- Requirement
  - Microsoft Visual C++ 2010 Express
    http://www.microsoft.com/visualstudio/jpn/downloads#d-express-windows-8
    (Sorry for this Japanese page, you can find the corresponding German page)

- Download and install OpenCV
  - http://sourceforge.net/projects/opencvlibrary/files/opencv-win/

- Add the following to "Path" in the environment system variables
  - *C:\opencv\build\x86\vc10\bin (If you install OpenCV directly under "C:\")*

NOTE: By default, Visual C++ works in 32-bit mode!

By referring to the above information and Web pages, please try to install OpenCV by yourself.

Source codes that I will present in this course work on OpenCV 1.X-2.X!
If you want to install OpenCV 3.X, you need some modifications.

OpenCV Installation (Linux)

Ubuntu (Debian base)

- Requirements
  - *Basic softwares 1*: build-essential build-dep opencv libqt4-dev libgtk2.0-dev pkg-config opencl-headers
  - *Image processing*: libjpeg-dev libopenjpeg-dev jasper libjasper-dev libjasper-runtime libpng12-dev libpng++-dev libpng3 libpnglite-dev libpngwriter0-dev libpngwriter0c2 libtiff-dev libtiff-tools pngtools zlib1g-dev zlib1g-dbg v4l2ucp
  - *Basic softwares 2:* python autoconf libtbb2 libtbb-dev libeigen2-dev cmake openexr gstreamer-plugins-* freeglut3-dev libglui-dev libavc1394-dev libdc1394-22-dev libdc1394-utils
  - *Video processing:* libxine-dev libxvidcore-dev libva-dev libssl-dev libv4l-dev libvo-aacenc-dev libvo-amrwbenc-dev libvorbis-dev libvpx-dev

- Download and install OpenCV
  - http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/
  - mkdir tmp; cd tmp
  - tar -xvzf /tmp/opencv-2.4.6.1.tar.gz
  - cd opencv-2.4.6.1
  - cmake -DBUILD_DOCS=ON -DBUILD_EXAMPLES=ON -DCMAKE_BUILD_TYPE=RELEASE -DWITH_TBB=ON -DWITH_GTK=ON\
    -DWITH_OPENGL=ON -DWITH_QT=ON -DINSTALL_C_EXAMPLES=ON -DWITH_OPENCL=OFF -DWITH_CUDA=OFF\
    -DWITH_OPENNI=ON -DWITH_UNICAP=ON -DWITH_V4L=ON -DWITH_XINE=ON
  - make
  - sudo make install
  - sudo ldconfig

Fedra (Red Had base)

- Requirements
  - gcc, g++, python, gtk+-devel, libjpeg-devel. libtiff-devel, jasper-devel, libpng-devel, zlib-devel, v4l2*, totem, xine, unicap-dev, autoconf, (maybe cmake?)

- Download and install OpenCV is the same to Ubuntu

MAYBE: ffmpeg is required for video processing

Not tested, because I use another version of OpenCV on linux ☺

OpenCV Installation (Mac)

- Requirement
  - Xcode
  - MacPorts

- Install OpenCV by MacPorts
  - sudo port install opencv

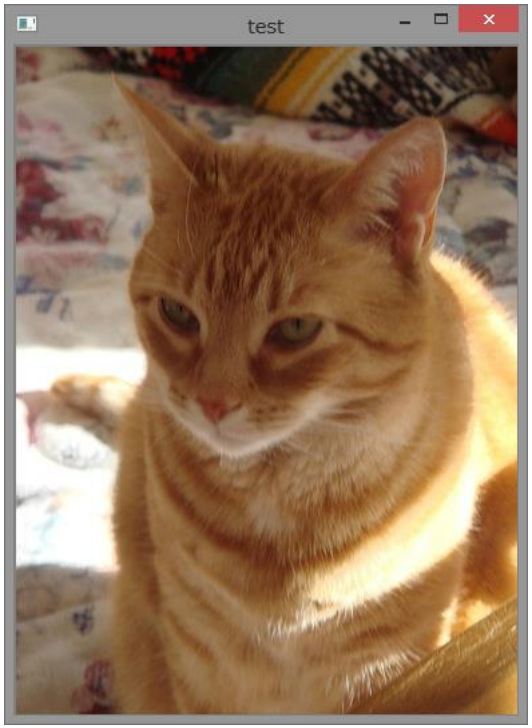
NOTE: You need your Apple ID to install Xcode

First program with OpenCV

- Read an image and show it in a dialog

```
int main(int argc, char* argv[]){
   IplImage* img = cvLoadImage("C:\\opencv\\samples\\c\\cat.jpg");
   cvNamedWindow("test", CV_WINDOW_AUTOSIZE);
   cvShowImage("test", img );
   cvWaitKey(0);
   cvReleaseImage(&img);
   return 0;
}
```

Please change the image filename depending on your environment.



IplImage is an old OpenCV structure to deal with images. But, in my opinion, rather than the recent cv::Mat, studying with IplImage is more useful for understanding how images are processed in a computer.

Windows Tips

- Setting the include directory of OpenCV

  Add C:\opencv\build\include to Project property -> Configuration properties -> C/C++ -> general -> Additional include directories

- Setting libraries of OpenCV

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

#ifdef _DEBUG
   // Debug mode
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_core246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_imgproc246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_highgui246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_objdetect246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_contrib246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_features2d246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_flann246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_gpu246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_haartraining_engined.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_legacy246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_ts246d.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_video246d.lib")
#else
   // Release mode
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_core246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_imgproc246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_highgui246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_objdetect246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_contrib246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_features2d246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_flann246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_gpu246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_haartraining_engined.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_legacy246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_ts246.lib")
   #pragma comment(lib,"C:\\opencv\\build\\x86\\vc10\\lib\\opencv_video246.lib")
#endif
```

Linux Tips

- Write include files at the beginning of your program

```
#include <cv.h>
#include <highgui.h>
```

- Complie

```
g++ -I/usr/local/include/opencv2 -I/usr/local/include/opencv -L/usr/local/lib -lopencv_highgui\
-lopencv_core -o <Runnable filename> <Source file>
```

(It is better to create makefile)

Mac Tips

1. Run Xcode

2. Create a project by selecting OS X -> Application -> Command Line Tool

3. Edit the following paths in “Build Settings (All)”
   - Add /opt/local/include to “Header Search Paths” in “Search Paths”
   - Add /opt/local/lib to “Library Search Paths” in “Search Paths”

4. Add OpenCV libraries to the project
   - Right-click the project folder, select “Add Files to…”
   and select libopencv*.dylib (short-cut files are not needed)

NOTE: You may find other OpenCV installation and usage which are better than this slide.
Please find the best way by yourself! If you have questions or troubles, please ask me.

Also, downloading required program for OpenCV installation may take very long time.
If the installation cannot finish during this lesson, you can install OpenCV at home.

The subsequent lectures assume that you can use OpenCV on your laptop.