Robot Vision Systems

PhD course spring term 2015

Michael Felsberg

michael.felsberg@liu.se

Goals

- System design and programming in
  - –OpenCV 3.0 rc1
  - –ROS (robot operating system) Indigo (Jade ?)
- focus part 1: visual computing
  - –basically same as OpenCV course 2012
  - –platform for own computational schemes and estimation methods
  - –own vision models and visual representations
- combining with high-level tools provided by OpenCV in the project part

Goals

- lectures concentrate on the fundamental data structures and how to manipulate and extend those
- focus part 2: robot vision systems
  - –distributed computing with ROS
  - –efficient use of OpenCV in ROS
- access to robotic hardware is not part of the course – make use of available resources from your lab
- simulation as fallback

Prerequisites

- solid background in
  - –mathematics (linear algebra, numerical methods)
  - –signal/image processing
  - –computer vision
  - –C++ programming
- own laptop with internet access and admin rights to install software
- camera supported by Ubuntu 14.04 (15.04 ?)

Organization

- lectures
  - –core features of OpenCV 3.0
  - –systems basics in ROS Indigo (Jade ?)
- seminars
  - –participants present topics
  - –one seminar presentation required for credits
- exercises
  - –installation of OpenCV and ROS
  - –going through essential first steps
- project (example application)

Organization

- credits: 9hp if
  - –project work
  - –80% presence
  - –one seminar presentation
- without the project work: 6hp
- note: if you have participated in the course ‘visual computing with OpenCV’, you can only get 6hp (3hp without project)
- 'listen-only’: 0hp

Schedule

- lecture 1: course information and OpenCV history. April 30, Thursday, 13.15 - 14.45
- exercise 1: installation of Ceemple: Eclipse, Python, and OpenCV. May, w19 Mon?
- lecture 2: dense matrices. May, w19 Wed?
- seminar 1 (topic 1-4). May, w20 Mon?
- lecture 3: methods on dense matrices. May, w20 Wed?

Schedule

- seminar 2 (topic 5-8). May, w21 Mon?
- lecture 4: sparse matrices and methods. May, w21 Wed?
- exercise 2: adding functionality to sparse matrices. May, w21 Thu?
- lecture 5: building your own representation. May, w22 Mon?
- exercise 3: build and test your own representation. May, w22 Wed?

Schedule

- lecture 6: good desgin principles, selection of projects. May, w22 Thu?
- lecture 7: rapid prototyping using Python. May, w23 Mon?
- exercise 4: Python: basics and prototyping using OpenCV. May, w23 Tue?
- lecture 8: debugging in OpenCV. May, 23 Thu?
- Part 2 (ROS) and final workshop (presentation of projects): Start in August, w33 or 34?

Seminars

1. classes (fundamentals)
2. classes (templates and namespaces)
3. vectors in STL (without iterators)
4. iterators in STL (mainly for vectors)
5. inheritance and virtual methods
6. exceptions
7. debugging: gdb
8. documentation with Doxygen

What is OpenCV?

- Open Source Computer Vision Library
- library of optimized algorithms (>2500)
- aimed at real-time computer vision
- developed by Intel, and now supported by Willow Garage and Itseez
- free for use under the open source BSD license
- cross-platform

History of OpenCV

- Intel Research Initiative
- Project launch 1999
- Related to Intel's Performance Library (today: IPP, Integrated Performance Primitives)
- Looking for CPU-intensive applications
- Project goals
  - –Advance vision research by open and optimized infrastructure
  - –Disseminate vision knowledge with readable code
  - –Advance commercial applications

Versions

- alpha-release at CVPR 2000
- five beta-releases 2001-2005
- Version 1.0 2006
- Continuation of development by Willow Garage 2008 (pre-release version 1.1)
- Version 2.0 2009
- Versions 2.1, 2.2 2010
- Version 2.3 2011
- Version 2.4 2012-2014
- Version 3.0 beta November 2014
- Version 3.0 rc1 April 2015

Applications

- 2D and 3D feature toolkits
- Egomotion estimation
- Facial recognition system
- Gesture recognition
- Human–computer interaction (HCI)
- Mobile robotics
- Motion analysis
- Object detection and recognition
- Segmentation
- Stereo vision: depth perception from 2 cameras
- Structure from motion (SFM)
- Motion tracking

Machine Learning

- Boosting
- Decision tree learning
- Gradient boosting trees
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)

Programming Languages

- Originally in C, since 2.0 also C++
- Wrappers to many other languages, a.o. Python, Matlab, and Java, although sometimes a bit outdated
- Since 2010 CUDA-based GPU interface
- Many desktop platforms (Windows, Linux, FreeBSD, OpenBSD, Mac OS)
- Mobile platforms (Android, Maemo, iOS)
- Primary vision package for ROS (Robot Operating System)

Why Using OpenCV?

- Many algorithms (>2.500)
- Efficient implementations
- De-facto standard (>7.000.000 downloads)
- Free to use
- Source code
- Quick bug-fixes
- Platform independent
- Rapid prototyping with Python

Decisions within Course

- ROS requires Ubuntu (Windows: VirtualBox)
- OpenCV option 1: Ceemple IDE (license will be provided)
  - Platform independent
  - Based on Eclipse IDE (integration of g++, gdb, svn, doxygen)
  - Includes OpenCV, Qt, OpenCL, Eigen, Boost, Dlib, etc.
  - Aims at replacing Python (rapid prototyping in C++)

Decisions within Course

- OpenCV option 2: Ceemple for VS
  - –Requires Visual Studio Community (Windows)
  - –Image Watch Extension and Project Wizard
  - –Includes OpenCV, OpenCL, etc.
  - –Qt etc might need to be installed separately
  - –Aims at replacing Python (rapid prototyping in C++)
- Python installation:
  - –Ubuntu: via package tool
  - –Windows: WinPython 3.4.3.2 (OpenCV 3) / 2.7.9.4 (OpenCV 2.4)

How does CVL use OpenCV?

- Alternative to Matlab with mex-files
- Collaboration with other labs
- Combined with ICE or ROS for building distributed real-time systems
- Connected to hardware APIs (e.g. LadyBug3)
- Undergraduate courses: project work

Motivation to give Course

- Used from Matlab: calculate with image data
- Images are matrices, thus entities in computations
- OpenCV 2/3 uses Mat for both images and matrices
- Support for doing calculations is limited, in particular on sparse data
- Gained knowledge on both cross-platform development and extending Mat-capabilities: to be shared!

Links

- http://www.ceemple.com/buy/ (license for course exists)
- http://sourceforge.net/projects/winpython/files/WinPython_3.4/3.4.3.2/
- http://www.robotappstore.com/Knowledge-Base/ROS-Installation-for-Windows-Users/137.html
- http://releases.ubuntu.com/14.04.2/ubuntu-14.04.2-desktop-amd64.iso
- http://download.virtualbox.org/virtualbox/4.3.20/VirtualBox-4.3.20-96997-Win.exe
- -install Ubuntu in a new virtual machine (16 GB)
- -resolution will initially be poor; install Guest Additions by clicking 'devices' in the Virtual Machine
- http://wiki.ros.org/indigo/Installation/Ubuntu

OpenCV without Ceemple

- Windows: http://www.cvl.isy.liu.se/education/graduate/opencv/opencv-installation-windows
- Mac: http://www.cvl.isy.liu.se/education/graduate/opencv/opencv-installation-mac-os
- Linux (Fedora): http://www.cvl.isy.liu.se/education/graduate/opencv/opencv-installation-linux