Stavanger, August 29, 2024

University of Stavanger

Faculty of Science and Technology

ELE610 Applied Robot Technology, H-2024

Image Acquisition assignment 2

For this assignment each group should write a brief report (pdf-file). Answer the questions and include figures and images as appropriate. You may answer in Norwegian or English, or a mix. The report may include new code you have created for your program `appImageViewer2X.py`. But you should also be prepared to run your program in the laboratory and show this to the teacher.

The intention here is that you should do as much as you are able to within the time limit for each assignment, which is 15-20 hours. A report containing a table showing time used, for each student of the group, will normally be accepted even if all tasks are not done. If all tasks are done, the time report does not need to be included.

2 Image acquisition using μEye XS camera

The Imaging Development Systems GmbH (IDS) μEye XS camera should be used in this assignment to capture an image, and to do some simple image processing on the image in Python.

Generally, the IDS web pages ↗ are good, but **access is restricted**, only registered users may access many of the technical pages. The registration is free, and not too complicated. Also the camera model is needed as a key to access product spesific information, we have `UI-1007XS-C` and `UI-3140CP-C`. The most useful, for us, parts of the IDS documentation are listed below.

- The uEye XS web page ↗ gives an overview of this camera. Among other things it says: This camera is supported by the IDS Software Suite SDK.
- IDS Peak is IDS new software for camera interface. The FAQ page ↗ says: For uEye cameras please note that in addition to IDS peak, the

Karl Skretting, Department of Electrical Engineering and Computer Science (IDE),
University of Stavanger (UiS), N-4036 Stavanger. E-mail: karl.skretting@uis.no.

latest version of the IDS Software Suite (4.95 or higher) must be installed. For more information and to operate uEye cameras in IDS peak applications, please refer to the IDS peak manual ↗.

- The IDS Software Suite web page ↗ tells that this software basically is obsolete. This page says, among other things, that IDS recommend a switch to IDS peak, but apparently this is not the complete thruth as we can read on the page link above. There is also a link to a download page for those who wants (needs) the old software.
  When IDS software (Software Suite) is installed, documentation (as html-files) is also stored on the computer, for my laptop they are located in `C:/ProgramFiles/IDS/uEye/Help/uEye_Manual/index.html`,
  or one my office PC `.../Downloads/uEye_Manual/index.html`.
- IDS Camera Manager, select the camera to use, if several are available, and display camera properties.
- uEye Cockpit, adjust camera parameters and capture image or video.
- Part C Programming describes the function that also can be used from Python through the `pyueye` interface.
- Part D Specifications gives a detailed specification of the camera.

2.1 Use IDS programs

IDS has updated their software and replaced the IDS Software Suite by IDS Peak. This is perhaps not completely done for old camera models as the ones we have, so I think old software is still needed but am not sure. Anyway, I have yet not installed IDS Peak as the old software works well on my laptop PC. However, I am very interested in your experience with IDS Peak and what is needed to sucessfully solve the tasks in these assignenments using the IDS Peak tools.

You may install IDS Peak from IDS web-page, but you probably have to look around and search until you find the download page. This done, follow the installation instructions and check that the software works from Windows menu, that an image can be caputured using an IDS camera. If this works the Python package should also work.

Perhaps you should also (or rether) install the IDS Software Suite as described below. You should install a working camera driver from IDS, the IDS Software Suite version 4.94 from July 2020 is ok. A newer version (checked June 2022) is IDS Software Suite version 4.95.2 (date 2022-03-23), will also do fine I guess, but version 4.96 doesn't work as well as it should according to some students that tried it. I suppose drivers are available on IDS downloads ↗, the release notes ↗ gives additional information. There are drivers available for Windows, 32 bit and 64 bit, and Linux. The same driver can be used for all IDS cameras. Only registered IDS users can download the drivers, the registration is free, and not too complicated.[1]

Install drivers and programs from IDS on your laptop, or use one of the laboratory PCs where this software is installed. Attach camera to USB-gate on the PC and start “IDS Camera Manager”. The attached camera should be visible in “Camera list” on the top of the program window. The buttons in the middle of the window will display general information and specific camera information. You may double-click on the line showing the camera to start the μEye Cockpit program. Especially note the help button which will start the useful “User Guide”. Try this and learn to know which sections are available, and use this guide whenever needed.

Back in μEye Cockpit program you can now try the different alternatives, behind each of the larger buttons. Try some of these, in particular investigate the different options that may be adjusted for this camera. Eventually, use the “Optimal Colors” button and capture and display image and video. The scene should include some of the colored dices that should be somewhere in the laboratory, perhaps on the camera rig table where they are supposed to be when not used. Save one image and include it in the report, on example is in figure 1.

2.2 Use IDS camera and Python

This section continues the Python section in assignment 1. In particular you need to install Python and all the needed packages. To do this, **carefully read** the instructions in the Fragments of Python stuff ↗ document. Here, the most important sections now are section 2.3 and section 4.2.

After installation you may continue directly with the list of tasks below. But, it will be helpful to start with the simple example in `appSimpleImageViewer.py` ↗ to get the basic ideas of GUI programming using Qt. Read from part 5 of the Fragments of Python stuff document, view the tutorial videos available from *canvas*, and perhaps also view some basice Qt tutorial videos or documents from www. Note that `appSimpleImageViewer.py` don’t use OpenCV, numpy or pyueye.

When you understand how `appSimpleImageViewer.py` works you can move on to `appImageViewer.py` ↗, and continue with `appImageViewer1.py` ↗ and finally `appImageViewer2.py` ↗. The one numbered 2 builds on the one numbered 1 by adding a Camera-menu to it. It will be helpful to view the videos

[1]If you dislike registration, students may copy the driver from another student or the teachers laptop (OneDrive or `../ELE610/IDS camera/`) or his USB memory stick (PNY).

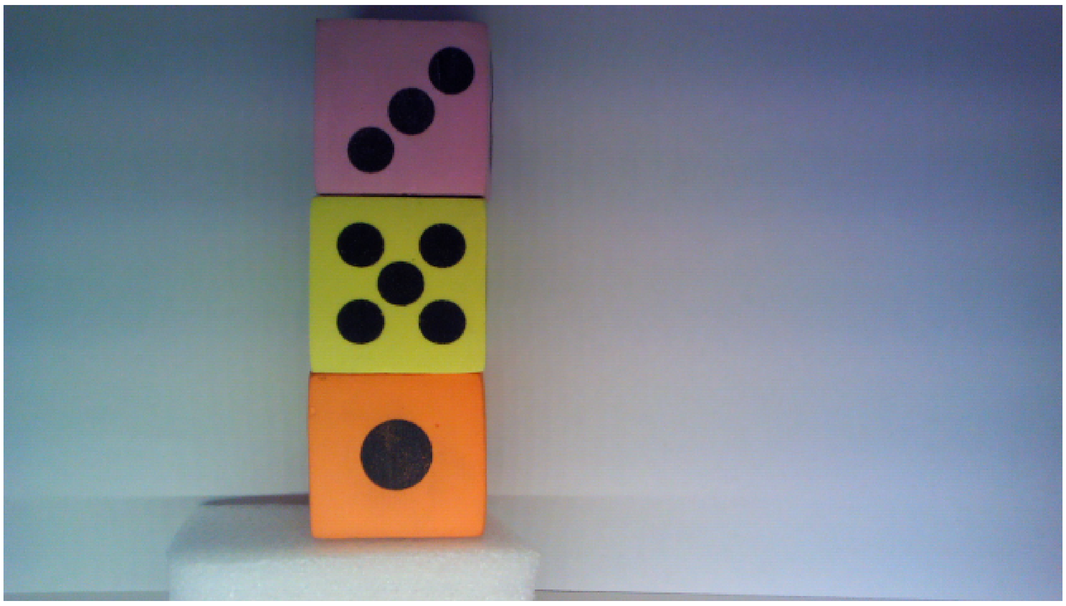
Figure 1: Image of dices captured using the uEye XS camera on UiS camera rig. The image is down sampled to size 640x360.

available from *canvas*.

Your solution may be done by understanding the code in `appImageViewer2.py` and then add the needed new features and perhaps even improve the features (partly) implemented. It is when you try to do some programming yourself or modify existing code that you really see how well you understand Qt and OpenCV. These examples are not complete, there is still some (much) work left to make your program `appImageViewer2X.py` better (perfect?). `X` should be replaced by your group letter.

a. This task does not need to be build into your `appImageViewer2X.py` solution. It may be helpful to see the example `pyueye_example_*.py` to better understand image acquisition. But this example is not easy to understand and may be completely confusing. Anyway, install `pyueye` and check that it works as intended.

   You may use the `cam_info()` function in `bf_tools.py` ↗[2]. Read the code in `cam_info()` carefully and try to understand it. In particular, look up the documentation for the used functions from the `pyueye` package; C++ versions of these functions are documented in IDS help system.

   You may also use a Python class that wraps around the IDS driver and thus makes it easier to use. This file is: `clsCamera.py` ↗. This file was written by Elias Nodland and Vilhem Assersen as part of their UiS Bachelor thesis May 2022.

[2]I am not yet satisfied with `bf_tools.py`, it will probably be improved, or deleted, later.

b. Try to understand what is done by the “Camera” menu items of `appImageViewer2.py`. To test that you understand this part you may make a new “Find focus”-function and place it before the “Get one image”-function in the menu. It has been discovered that focus is found after 15 to 25 images are captured, so the simple implementation is just to capture some images and return assuming autofocus has worked. The “Find focus”-function could also be called from (near the end of) the cameraOn-function.

   An alternative is to look at the difference between two following images captured, and when this is small it indicate that the focus is found. You could try to make a function that works well for different scenes, but most important is the scene where dices are 200 to 400 mm from camera. Include this function in `appImageViewer2X.py`.
   WARNING: It is easy to spend lots of hours on this point, but try not to do this. Two hours should be enough, at least to make the simple solution work.

c. Add at least one more point to the “Camera” menu. Perhaps print more camera information, perhaps capture two images just after each other and find the difference between the two images, subtract the first from the second image. More difficult tasks are to change the camera options (not all are possible to change) and to capture a video sequence.

d. Add another menu `Dice` (in `appImageViewer2X.py`) and under it add an Action for `Black dots`. You should try to find the black pixels using “toBinary”-example in `appImageViewer1.py`. Morphological functions can remove noise and only keep the eyes as the black dots. Do not make the task more difficult than necessary, use only one dice and have a smooth background without black areas.

e. You may also add an action (to `appImageViewer2X.py`) named `Find circles` that uses `cv2.HoughCircles()` function and is an alternative to `Black dots`. This action/function is used in `appImageViewer3.py` and you may look there for hints. But it should be added to your program `appImageViewer2X.py`.

f. Add a feature, an action, `Count eyes`, in the `Dice` menu. This should find the number of eyes in the captured image after black dots (or circles) are found. This should also be in `appImageViewer2X.py`.

g. And finally, if you still have time left, make your solution `appImageViewer2X.py` even better. It may capture images continuously (video) or regularly (each second) and print out the new state each time the number of eyes have changed. Correct any errors, clean up any messy part, make comments sufficient, but not too verbose.