Practical Robotics
Exercise 3

Marc Toussaint, Yoojin Oh
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

April 24, 2019

1 OpenCV installation

Please update the repo, using `git pull; git submodule update`. Ensure that libopencv-dev is installed with ubuntu. (E.g., in rai/rai/Perception, try `make installUbuntu`) Then recompile using `make dependAll; make -j4`.

The example in `cpp/p2-cameraMini` subscribes to the camera and starts a little loop that displays the image using opencv. In this exercise you should fill this loop with opencv code (directly operating on `cv::Mat`) to perform some basic image processing.

The default installation of opencv only runs in python2. The default install is `sudo apt-get install opencv-python`. Before importing `import cv2` you might have to remove a ros path, namely `sys.path.remove('/opt/ros/kinetic/lib/python`

However, in some jupyter notebooks we could not run python2 (even when starting a python2 kernel `sys.version` would show python3). To use python3 you need to install a newer opencv version from source. Instructions are here `https://github.com/MarcToussaint/rai-maintenance/blob/master/help/localSourceInstalls.md` – but not tested how to use this from within jupyter.

2 OpenCV

You can do these exercises also without having installed OpenCV yet. But eventually you'll need install the above and use the images published in ROS to be able to work also with Baxter.

There are many OpenCV tutorials:

- For python, see `https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html`
- In C++, see `https://docs.opencv.org/2.4/doc/tutorials/tutorials.html` (perhaps change to another OpenCV version)

Esp, see "GUI features - Videos" if you want to develop your code with your webcam, and "Image Processing".

Exercise 1:

1. Filter the color of the image to find all pixels that are redish.
2. Display the binary image that indicates red pixels.
3. Fit contours to this binary image, which returns the segments.
4. Display the contours by drawing them into the original RGB image

Exercise 2:

1. Store the first image as background image. (Or average over several first images.)
2. For every new image filter those pixels, that are significantly different to the background.
3. Display the binary image that indicates change pixels.

4. Fit contours to this binary image, which returns the segments.
5. Display the contours by drawing them into the original RGB image
6. Test the same pipeline, but first smoothing/blurring the image

3 Project Proposal Draft

Start setting up a document that sketches your project proposal. The document should include short descriptions of the goal, problems & methods, plan & milestones, requirements.

– Goal ("what"): Describe in detail the behavior you want so see on the robot. This section *must not* make any statements on what method you are using to create this behavior. E.g., saying "using Reinforcement Learning ..." is not a goal description! Only describe the behavior we can eventually see on the robot, as detailed as possible.

– Problems and methods ("how"): Describe how the goal can be achieved by providing a list of sub-problems to be solved. List methods (algorithms, libraries, potentially several alternative ones) to solve these problems. E.g. "OpenCV, with filtering based on color to find the object", etc

– Plan and milestones: Outline a rough timing and work plan. What will you address in which order? What are core milestones (usually demonstrations of parts of the behavior)?

– Requirements: What are additional requirement for your project, e.g. from us (providing other interfaces or hardware or materials).