Assignment 1

Robot Vision practical

Friedrich Fraundorfer (fraundorfer@icg.tugraz.at)

Deadline: April 24, 2024, 23:55h
Questions: If you have any questions about the exercises send a mail with subject prefix: "[RV KU]" to marco.pfleger@student.tugraz.at.

Submission: Submit the completed assignment (code, output data and report) using the TeachCenter submission system as a **single** ZIP File. Documents need to be submitted using the PDF format.

Compulsory submission file structure:

```
assign1.zip/                                  root directory
    └ code/                                   contains all code files
        └ assign1_task1.cpp
        └ assign1_task2.cpp
        └ assign1_task3.cpp
        └ assign1_task4.cpp

    └ report/                                 contains a single PDF
        └ assign1_report.pdf
    └ fileoutput/                             stores all relevant output
```

Task 1: Camera calibration (5pts)

This task consist of writing a program for computing the individual camera calibration parameters from the provided image sequence of a calibration pattern (**CALIB_DATA.ZIP**). The following parameters need to be estimated and reported for each camera:

- fx, fy, cx, cy
- k1, k2, p1, p2, k3

The physical size of a black (white) patch in the calibration image is 30x30mm and the pattern has 8x5 patch intersection.

Deliverables:

Document the estimated camera parameters in the report. Document also a quality report of the calibration (overall RMS re-projection error). No file output is required.

Hint:

Follow the tutorial at https://docs.opencv.org/master/d4/d94/tutorial_camera_calibration.html. It is not necessary to write totally new code. You can start with the provided example of OpenCV and modify this to your needs. Important, the estimated camera parameters are needed in Task 2.

Task 2: Stereo calibration (10pts)

This task consists of writing a program for calibrating a stereo setup from the provided images of a calibration pattern (**CALIB_DATA.ZIP**). The following parameters need to be estimated for the stereo setup:

- R … Output rotation matrix between the 1st and the 2nd camera coordinate system
- t … Output translation vector between the coordinate systems of the cameras
- e … RMS of re-projection error (as quality measure)

Deliverables:

Document the estimated calibration parameters in the report. No file output is required.

Hint:

Follow the example stereo_calib.cpp from the OpenCV installation. It is not necessary to write totally new code. You can start with the provided example of OpenCV and modify this to your needs. Important, the estimated calibration parameters are needed in Task 3.

Task 3: Stereo matching (10pts)

This task consists of writing programs for performing stereo matching. Your submission should take as input two images (from **STEREO_DATA.ZIP**), intrinsic calibration (from Task 1) and extrinsic calibration (from Task 2) and perform stereo matching. The submission should output a disparity image and write out the depth data as colored 3D point cloud in the PLY format. The output should be generated with the Unimatch method. Proper parameters for stereo matching should be chosen, such that a meaningful result is obtained.

The Unimatch method can be downloaded from:

https://github.com/autonomousvision/unimatch

Unimatch comes with a variety of differently trained models. For this assignment the model "GMStereo-scale2-regrefine3-resumeflowthings-middlebury" is to be used. Unimatch is a Python repository. The interface between the OpenCV part and the Python stereo matching can happen using file-IO.

Deliverables:

Put the disparity image, the rectified input images and a screenshot of the visualized point cloud (e..g Meshlab) into the report. Three file outputs are required. The disparity image as PNG, the rectified image pair as PNG and the colored 3D point cloud as PLY.

[Optional: Process also the additional stereo pairs from **STEREO_DATA_ADD.ZIP**.]

Hint:

Follow the example stereo_match.cpp from the OpenCV installation. It is not necessary to write totally new code. You can start with the provided example of OpenCV and modify this to your needs to create an interface to Unimatch.

Task 4: Measuring in-image object dimensions using depth data. (5 pts)

Utilizing the depth data that has been obtained from the previous task, please estimate the dimensions (width, height and length in meters) of the "Stochastic Process and Filtering Theory" book seen in the provided stereo image data.

Deliverables:

The dimensions must be mentioned in the report and the report needs to show where the measurements have been taken (e.g. showing the RGB image with measured points marked) and needs to describe the measurement process.

Additional information:
The tasks must be solved using the OpenCV library using C/C++. It has been tested with **OpenCV 4.9**.

Important: Make sure to install opencv-contrib including the non-free module.

https://docs.opencv.org/master/d7/d9f/tutorial_linux_install.html

Auf Option: OPENCV_ENABLE_NONFREE=ON achten

https://docs.opencv.org/3.4/d2/dca/group__xfeatures2d__nonfree.html

OpenCV works fine under Ubuntu run in Virtual Box (https://www.virtualbox.org/).

OpenCV can be downloaded from:

https://github.com/opencv

Tutorials can be found at:

https://docs.opencv.org/master/d9/df8/tutorial_root.html