IMC-5204B – Computer vision
Lab 2 (TP) : camera calibration

Xavier Hilaire
x.hilaire@esiee.fr

21 novembre 2019

IMPORTANT

- This lab is supposed to be ran on Linux Debian 9, using OpenCV 2.4, as provided in ESIEE's lab rooms (5201, 5207). Your submission will be tested on ESIEE's machines, so it must work there.
- Wwork alone, or by pairs.
- Write your answers on a hard copy, and give it to me at the end of the next lecture, schedulded on **6th of december 2019**.
- Create a `TP2` directory and store all the files you are working with there. Save your answers to the questions in an `answers.txt` file in that directory.
- Please, do not specify absolute, but relative paths to your images or files in your code.
- Create a ZIP archive of your `TP2` directory, and upload it thanks to this form : `https://mvx1.esiee.fr/tp_drop.html`. Please, no submission by email (-1 pt penalty for not observing this).

1 Hardware setup

Open a session on Linux. In a terminal, cd to your `TP2` directory, then launch the following command :

```
curl https://perso.esiee.fr/~hilarex/IMC-5204B/TP2.tgz | tar xvpzf -
```

Plug **one** camera only out of the 2 you have been given in a USB port. Launch `xawtv` from the terminal to make sure it is working.

2 Estimating the camera instrinsincs using Zhang's method

The `calibration.cpp` program demonstrates how to calibrate a camera using Zhang's method. The calibration pattern is an 8×5 chessboard, each square being $3cm \times 3cm$ wide. You should therefore run the program with something like

```
./calibration -h 8 -w 5 -s 3.0 -op
```

As you press the 'g' key, the program will take 10 shots of the chessboard, calibrate the camera, and save the intrinsics in a `out_camera_data.yml` file.

1. You should show the chessboard to the camera in different orientations. Why ?
2. Try to figure out broad values for the α and β intrinsics. You will need a ruler to measure the distance from your chessboard to the camera.
3. How do you interpret the values given in this matrix ? In particular, compare α and β to what you estimated before.
4. What can you say about the distortion coefficients ?
5. When it returns, the `calibrateCamera` functions provides the translation and rotation vectors to each view of the calibration pattern you acquired during calibration. Refer to `http://docs.opencv.org/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html#calibratecamera` for more explanations. Modify `calibration.cpp` so that it also print these rotation and translation vectors (look for +XH/-XH tags in the code).
6. According to you, in which coordinates referential are these vectors expressed ? What is the main drawback of Zhang's calibration method ?

Make a backup of your files – very important : they might be overwritten hereafter if you don't follow the instructions exactly.

3 Estimating the orientation and location of the checkboard

3.1 New shots

The `calibration.cpp` programs also saves greylevel versions of the images used for calibration as `calib`*xx*`.pgm` files, and creates some `points`*xx*`.txt` files. The latter files contain the (u, v) coordinates of the image points of the corners of the chessboard acquired during calibration. Choose any of them and plot it (using gnuplot or whatever software you like).

Modify `calibration.cpp` so that it delays for a longer time between 2 shots.

In a separate directory take at least 7 shots of the pattern, roughly as follows :

— Pure rotation : while trying to keep the center of the chessboard aligned with the optical axis of the camera, and at a constant distance of it : One shot parallel to the image plane, one with an angle of about 45^o vertically, one with an angle of about 45^o horizontally, one with angles of about 45^o both horizontally and vertically

— Pure translation : while trying to keep the center of the chessboard broadly aligned with the optical axis of the camera and its orientation unchanged : one shot at a given position, one shot 50cm further, one shot 1m further. (you may try translations alongst horizontal/vertical axes as well).

At this point, backup your directory (everything) to a `stage1.tgz` file.

3.2 Estimating the orientation

3.2.1 Formulas

Consider the 4 "corners" A, B, C, D of the convex hull of the image points you obtained for a chessboard in general position (the plane has undergone 2 rotations, not only one), as shown in Fig. 1. Let (u, v) be the camera coordinates of the vanishing point due to parallel lines $(AB), (DC)$, and similarly, let (u', v') be those for the parallel lines $(AD), (BC)$.

1. Show that if the chessboard lives in a plane which is not parallel to the image, then the vector with world coordinates

$$\mathbf{n} = \begin{pmatrix} \alpha(v - v') \\ \beta(u' - u) \\ uv' - vu' \end{pmatrix} \tag{1}$$

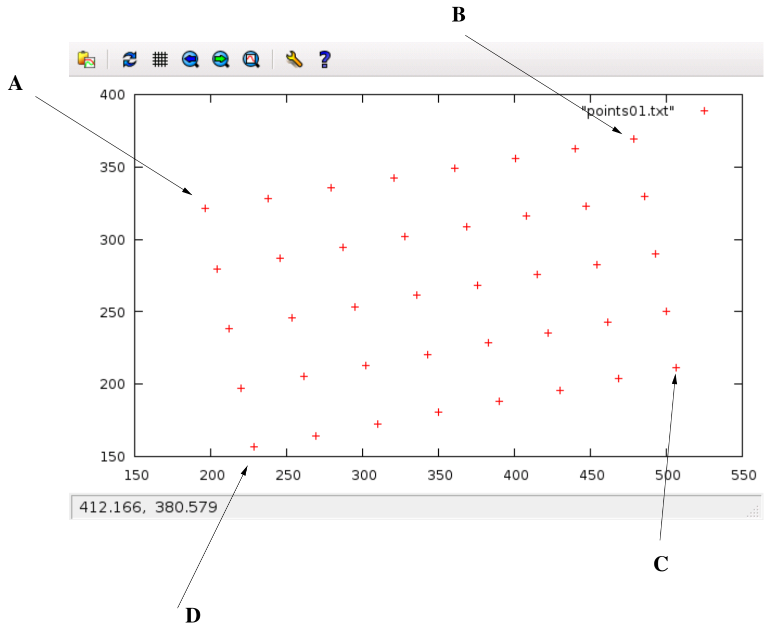

FIGURE 1 – Extracted corners

is normal to the chessboard plane.

2. Given that the azimuth ϕ and meridian θ angles of any vector with world coordinates (a, b, c) are given by

$$\phi = \arctan\left(\frac{c}{\sqrt{a^2 + b^2}}\right), \tag{2}$$

$$\theta = \arctan\left(\frac{b}{a}\right) \tag{3}$$

express ϕ and θ as functions of the coordinates of A, B, C, D.

3.2.2 Test program

Write a C (or C++) program `angles.c`, that takes the filename of any of your .txt point files as argument, and evaluates ϕ and θ according to the formulas you found above. Test it at least on the images you acquired during calibration, and possibly on a separate image set (beware to do this in a distinct directory so as not overwrite your work!)

Note that although (1) is theoretically correct, you might observe not so good results in practice. Can you guess why? Could you think of a way to improve them?

At this point, backup your directory (everything) to a `stage2.tgz` file.

3.3 Estimating the translation

Bonus, optional. Describe a simple procedure to get the position of the chessboard in real world coordinates. You are free to choose the reference point as you like (it can be a corner, or the center, for instance). Your method may reuse the orientation estimates (ϕ, θ) found before, or not.

Modify your `angles.c` so that it also displays the translation.

At this point, backup your directory (everything) to a `stage3.tgz` file.