TTK4255: Robotic Vision

Homework 6: Camera calibration

Instructions

For more information about the course work and how to get help, see the `Course work/about.pdf` document on BB. To get your assignment approved, you need to complete 60% (sum of task weights). Upload the requested answers and figures as a single PDF. You don't need to submit your code. You may collaborate with other students and submit the same report, but you still need to upload it individually on Blackboard. Please write your collaborators' names on your report.

Relevant reading

The camera calibration method that you will consider in this assignment, and use yourself in the final project, is explained in Matlab's tutorial (link) and associated text (link), which should be helpful even if you are not familiar with Matlab. The method relies on a planar object with a known pattern, and is commonly attributed to Zhengyou Zhang's 2000 paper, available on Blackboard. Implementations of Zhang's calibration method are available in the Computer Vision Toolbox (Matlab) and OpenCV (Python). A very brief overview of camera calibration can be found in Szeliski (2010) 6.3 and 6.4.

The Brown-Conrady distortion model

The pinhole camera model neglects many aspects of real cameras, most notably the presence of lenses, that can cause significant deviation from the theoretical pinhole camera projection. In modern cameras, the distortion caused by lenses is often small enough that it can be corrected by an appropriate non-linear transformation. Transforming an image with lens distortion into one that satisfies a pinhole camera projection is called undistortion.

Note: Although "lens distortion" is often presented in textbooks as an optical imperfection that needs to be corrected, it should be emphasized that a non-pinhole projection can be an intentional aspect of the lens design, and that many computer vision algorithms are trivially applicable—or have been extended to work—with non-pinhole projections. For instance, distortion is "used" in fisheye cameras to map a large field of view, often near 180 degrees, onto a planar imaging sensor, while providing a more evenly distributed pixel density per unit viewing angle.

There are many extensions of the pinhole camera model that can capture small-to-moderate lens distortion. One such extension is the Brown-Conrady distortion model. Let $x = X/Z$ and $y = Y/Z$. Then, in the pinhole camera model with Brown-Conrady distortion, a 3D point with camera coordinates (X, Y, Z) will be projected to the pixel coordinates

$$u = c_x + f_x(x + \delta_x), \tag{1}$$
$$v = c_y + f_y(y + \delta_y), \tag{2}$$

where the newly introduced scalars δ_x and δ_y contain two types of distortion, radial and tangential, which are modeled as polynomials in x, y and $r = \sqrt{x^2 + y^2}$:

$$\delta_x = (k_1 r^2 + k_2 r^4 + k_3 r^6 + \cdots)x + 2p_1 xy + p_2(r^2 + 2x^2), \tag{3}$$
$$\delta_y = \underbrace{(k_1 r^2 + k_2 r^4 + k_3 r^6 + \cdots)y}_{\text{Radial}} + \underbrace{p_1(r^2 + 2y^2) + 2p_2 xy}_{\text{Tangential}}. \tag{4}$$

The distortion coefficients $k_1, k_2, ..., p_1, p_2$ are estimated via calibration, along with c_x, c_y, f_x, f_y. We have also introduced $f_x = s_x f$ and $f_y = s_y f$, because we cannot estimate both the pixel densities and the focal length, due to them being multiplied. More tangential distortion parameters can also be used, but for modern cameras it is very often limited to two. This model, which is also called the radial-tangential or "radtan" polynomial distortion model, can be considered a standard, and is supported by many camera calibration tools, including OpenCV's calibration module (link), Kalibr (link) and Matlab's Computer Vision Toolbox (link).

The Brown-Conrady distortion model has proven to work well for modern cameras with a tele-, normal or wide-angle lens (field of view up to 120°). However, as you will explore in one of the tasks, it starts to become problematic when the field of view approaches 180°. For such cameras, one may choose to not undistort the image into a pinhole projection, and instead represent locations in the image by 3D unit bearing vectors and use algorithms that have been generalized for bearing vectors. If you are curious, you may check out OpenGV (link) for a collection of such generalized algorithms, but this is not part of the curriculum.

Part 1 Various theory questions (75%)

Task 1.1: (10%) If the 3D point to be projected is represented with spherical coordinates around the camera origin, then its projection in the (unextended) pinhole camera model can be written as

$$u = c_x + f \tan\theta \cos\phi, \tag{5}$$
$$v = c_y + f \tan\theta \sin\phi, \tag{6}$$

where θ and ϕ are the spherical coordinates, related to the point's camera coordinates (X, Y, Z) by

$$X = \lambda \sin\theta \cos\phi, \tag{7}$$
$$Y = \lambda \sin\theta \sin\phi, \tag{8}$$
$$Z = \lambda \cos\theta. \tag{9}$$

Use this result to explain why it can be problematic to attempt to undistort images taken by ultra-wide-angle cameras (field of view approaching $180°$) into a pinhole projection. Tip: What range do you expect for θ if the field of view is $180°$ (i.e. the camera captures a whole hemisphere)?

Task 1.2: (10%) Consider the pinhole model with three radial- and two tangential distortion coefficients. Assume the model has been calibrated on images of width W and height H. Consider an image taken by the camera and suggest how all the parameters $(c_x, c_y, f_x, f_y, k_1, k_2, k_3, p_1, p_2)$ should be adjusted to remain valid, when...

(a) ...the image is downscaled by 1/2 to width $W' = W/2$ and height $H' = H/2$.

(b) ...the image is cropped to width $W' = W - (l + r)$ by subtracting $l \geq 0$ pixels from the left and $r \geq 0$ pixels from the right.

Task 1.3: (5%) The images in Fig. 1 have been artificially modified to exhibit strong radial distortion. Suppose the distortion can be corrected satisfyingly with just one radial distortion coefficient (k_1). For which image (left or right) will k_1 be positive and for which will it be negative?

Figure 1: Images with simulated radial distortion.

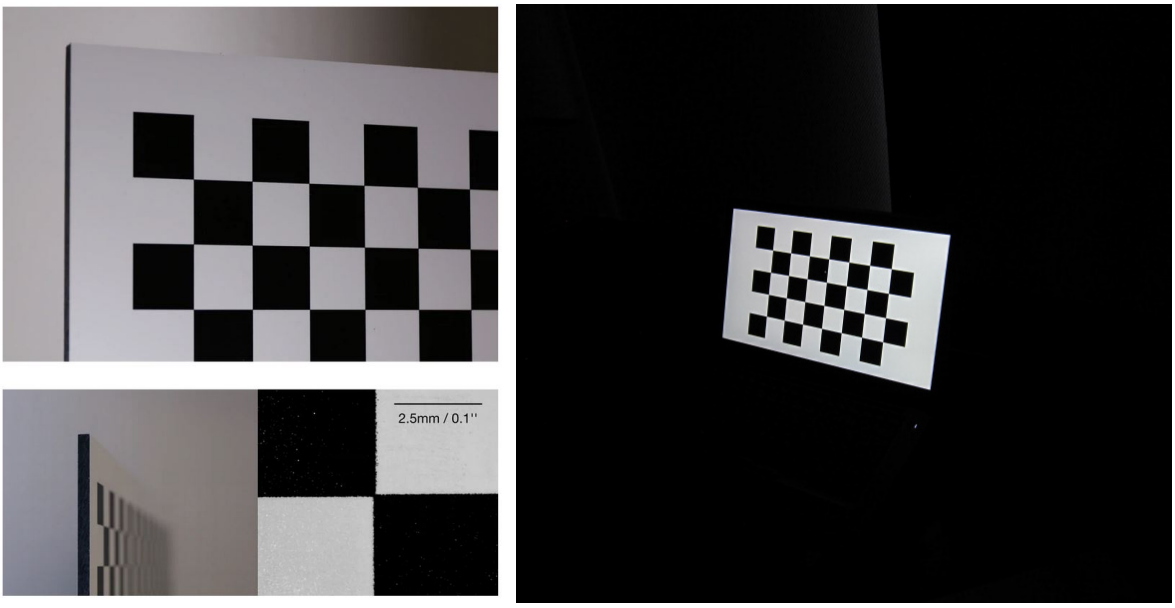

Figure 2: Left: Checkerboard pattern printed on a precision-manufactured aluminium plate, with a precise square size of 2.5 millimeters. Source: Calib.io (link 1) (link 2). Right: A checkerboard pattern displayed on a laptop screen, with the lights off and curtains shut to prevent glare.

Task 1.4: (5%) Camera calibration tools distinguish between intrinsics, which are assumed to be the same in every image used in the calibration, and extrinsics, which can (and should) vary from image to image. In Zhang's calibration method, the extrinsics are the poses (rotation and translation) of the camera relative to the calibration object. Suppose you have N calibration images. How many parameters must be estimated, assuming you use three radial- and two tangential distortion coefficients?

Task 1.5: (15%) Zhang's calibration method requires you to take images of a calibration pattern from various orientations. A common pattern is the checkerboard (Fig. 2), but of most importance is that the pattern has a number of points with precisely known coordinates. This raises the problem of ensuring that the real-world pattern precisely matches its digital representation.

One way to obtain a precise pattern is to order one from services like Calib.io. However, a far cheaper and easier method is to display the pattern on a computer screen, as these are usually sufficiently flat and sufficiently high resolution. You may fear that this will be imprecise due to the need to measure the metric scale of the pattern on the screen (e.g. the square size in the checkerboard). However, if you only care about the intrinsics, then you don't need to care about this. In the case of the checkerboard, you can simply specify an arbitrary square size. Explain why. (*Note: For checkerboard patterns it is always assumed that every square has the same size.*)

Task 1.6: (15%) Reprojection error is often used when assessing the quality of a camera calibration. However, reprojection error alone is insufficient to assess the accuracy of the parameters. Explain with an example how a calibration can give a low reprojection error, possibly exactly zero, yet likely be useless.

```
Focal length and principal point          Distortion coefficients
---------------------------------         ---------------------------------
fx:   2359.40946 +/- 0.84200              k1:     -0.06652 +/- 0.00109
fy:   2359.61091 +/- 0.76171              k2:      0.06534 +/- 0.00624
cx:   1370.05852 +/- 1.25225              k3:     -0.07555 +/- 0.01126
cy:   1059.63818 +/- 0.98041              p1:      0.00065 +/- 0.00011
                                          p2:     -0.00419 +/- 0.00014
```

Figure 3: Example calibration result.

Task 1.7: (15%) When collecting images for Zhang's calibration method, it is good practice to capture images of the pattern at an angle, such that the images exhibit perspective foreshortening. It is fine if the pattern appears frontoparallel (i.e. perpendicular to the optical axis) in some images, but it is problematic if it appears frontoparallel in every image. Use the pinhole camera model equations to explain why. You may assume that there is zero distortion.

Hint: Some of the extrinsics will become "confused" with some of the intrinsics.

Part 2 Analysis of an example calibration (25%)

Figure 3 shows an example calibration result that you may get from Matlab's Camera Calibrator App (link) or OpenCV's calibration module. The summary lists each parameter's estimated value and its standard deviation. The standard deviations indicate the uncertainty in the estimated parameters, and must always be considered in addition to reprojection error when assessing the quality of a calibration. Multiplying a standard deviation by 1.96 gives the half-width of a 95%-confidence interval (link). This confidence interval will then be a likely range to contain the true value.

It should be intuitive how uncertainty in the principal point translates into uncertainty in the image; in the projection of a given 3D point, any error in (c_x, c_y) results in an equal offset in (u, v). What may not be intuitive is the effect of uncertainty in the "focal lengths" and distortion coefficients. It should be clear that the resulting image offset will not have the same magnitude everywhere in the image. For example, if the 3D point lies on the optical axis, then it will always be projected to the principal point, regardless of the focal length and distortion coefficients (and, thus, any error in their estimates). By inspecting the pinhole equations, you should see that an error in these parameters has the largest effect on (u, v) when $r = ||x, y||$ is greatest, which is when the projected point appears near a corner.

Task 2.1: (10%) For the example calibration results above, and assuming that the image has width $W = 2816$ pixels and height $H = 2112$ pixels, calculate how large the offset in the horizontal pixel coordinate can be, if the error in f_x is 1.96 standard deviations. Assume that the other parameters have zero error.

Task 2.2: (15%) The tangential distortion coefficients, p_1 and p_2, are often omitted if their effect on the projection is negligible, e.g. less than one pixel in magnitude near the image corners. Can they be omitted in this case? Support your answer with calculations involving the estimated values for p_1 and p_2, and their standard deviations.

Tip: The provided script `task22.py/m` implements the camera model with distortion. You may use the script as aid in your calculations.