CS 223b: Introduction to Computer Vision
Assignment 1: Camera Calibration

Due date: **Thursday, January 13th 23:59 PST**
You may work in **teams of up to 3 persons**
Submission via email with subject "Assignment 1" to dahlkamp@stanford.edu

1 Camera Calibration (20 pts)

1.1 Software Installation (0 pts)

- Install the Intel `OpenCV` Library from `http://sourceforge.net/projects/opencvlibrary/` on your MS Windows computer. Although working with the Linux version is possible, experience shows that getting a webcam to work in Linux and OpenCV is a lot more work and therefore we do not recommend it in the beginning.

- Install a C++ compiler. MS Visual Studio 6, for example, is available in the Stanford CS Department Software Library, Gates Room 161.

- Compile and run the sample project from `OpenCV/samples/c/cvsample.dsp` to test that your machine is set up properly. It should open two windows showing a demo.

- Get a "Creative Vibra WebCam" camera, either during thursday's lecture, during CA office hours (Tue, Wed 9-11am Room 292) or from Fry's Electronics. We have 35 cameras available, which is one per two students. Install the camera driver to your Windows computer. In order for the camera to work with OpenCV, you will probably need to use the driver the autostart setup program installs, not the one Windows finds on its own.

1.2 Calibration Image Acquisition (5 pts)

- For calibration, OpenCV implements the [Zhang 99] algorithm. This algorithm inputs correspondences between 2D image points and 3D scene points over a number of images and outputs the intrinsic camera matrix as well as the radial distortion coefficients k_1 and k_2. It also computes the rotation and translation of the scene towards the camera for every image, but this information is not needed for calibration.

  The scene points of the correspondance have to be in the same plane and to obtain good results, this plane should be different for every calibration image.

  In practice, an easy way to achieve this is to print a chessboard pattern onto a piece of paper, attach this paper to a rigid surface and take the corners between the chessboard fields[1] as the correspondance points. Prepare such a calibration pattern.

- Write a program using OpenCV to open the camera video stream and grab a couple of images of the calibration pattern. To have reproducible results, it is a good strategy to first acquire and save some images and then load them from disk for the actual calibration.

  In OpenCV, you will find the following functions interesting:

[1] that is, use only the corners on the *inside* of the chessboard

  - `cvCreateImage` initializes an image data structure.
  - `cvNamedWindow` creates a window.
  - `cvShowImage` displays an image in a window.
  - `cvLoadImage` and `cvSaveImage` load and save image files.
  - `cvCaptureFromCam` initializes a video stream.
  - `cvQueryFrame` grabs a single frame from the camera.

  A complete function reference can be found at `OpenCV/docs/ref`.

- The Vibra Webcam has a dial on the front to change camera focus. As the cameras might have some manufacturing variation, please create two sets of images, one with the calibration pad in focus and one with the dial turned to the rightmost.

- Put the calibration pictures into your solution email.

1.3 Camera Calibration (15 pts)

- Create the set of correspondence points for each of your images. For the 3D scene coordinates, measure in inches and use (0,0,0) as coordinates for the top left corner. In the image, measure in pixels using (1,1) as the top left corner. Add these correspondence points to the solution email.

  Initially, you might want to choose the image coordinates manually to make debugging easier.

- Use the function `cvCalibrateCamera` to calibrate your camera using your correspondences. Add the intrinsic camera matrix and the lense distortion parameters to your solution email. Format floats using `printf("%4.1f")`.

- As you might have noticed, manually selecting the corner pixels in the image is tedious and we want to automate it. Use `cvFindChessBoardCornerGuesses` and `cvFindCornerSubPix` to find and refine these positions automatically.

Finally, please state the percent efford each individual member of your team has contributed to this assignment (numbers should add up to 100%). Attach your sourcecode and submit everything with subject "Assignment 1" to dahlkamp@stanford.edu.

You may keep the webcam if you like.

References

[Zhang 99] Z. Zhang: *Flexible Camera Calibration by Viewing a Plane from Unknown Orientations.* International Conference on Computer Vision (ICCV'99). Corfu, Greece, September 1999, pp. 666-673.