CS 223b: Introduction to Computer Vision
Assignment #2: Camera Calibration
(out of 25 points)

Due date: Monday, January 26th, 2003 (no extension)
N.B.: You are not allowed to work in teams on this assignment.
Recommended readings: *Trucco and Verri* Chapter 6

1 Calibration with MATLAB (20 pts.)

(a) **(20 pts.)** You are given the locations of known points:

http://robots.stanford.edu/cs223b/homework/hw2/points3d.txt

on a 3D calibration target depicted below in Figure 1. Your job is to create a MATLAB function of the

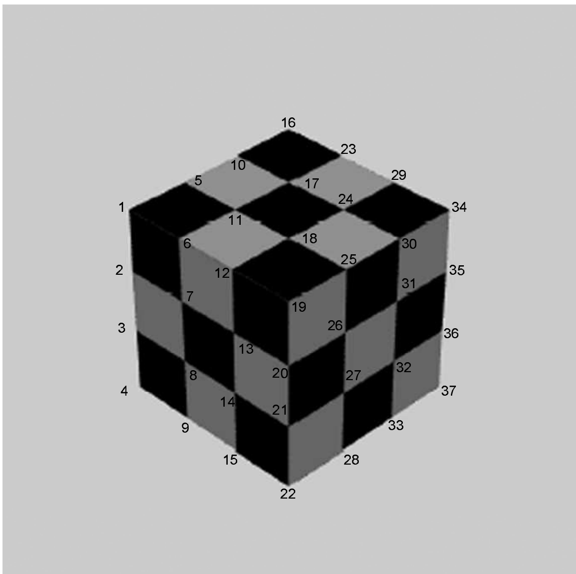

Fig. 1: Three dimensional calibration phantom.

form:

function [R,T,fx,ox,oy,alpha] = calibrate(pts3d, pts2d);

which takes as input the $[37 \times 3]$ matrix of 3D point locations together with their corresponding 2D image locations (in [column, row] format) and derives the camera's calibration parameters.

Specifically, compute:

- (i) the rotation matrix R and translation vector T in equation (6.1) from the textbook
- (ii) the horizontal focal distance f_x (focal length in effective horizontal pixel size units)
- (iii) the image center (o_x, o_y)
- (iv) the pixel aspect ratio α, assuming the image is free of distortion.

What you need to do:

1. Download the original calibration image from the class website:

   http://robots.stanford.edu/cs223b/homework/hw2/calib.pgm

2. Localize the feature corners in the image *in [column, row] format*. You may **either** do this manually or utilizing your previous corner-finding code (**N.B.** you will almost certainly have to make changes to your code for this real world example.) You should write a function called:

   function pts2d = localizeCorners(Im);

   Which takes the calibration image matrix as input and returns the $[37 \times 2]$ matrix of locations of its corner pixels *in [column, row] format*. If you choose to localize these pixels manually, just hard code your values into this function.

3. Using **localizeCorners** and following Trucco and Verri, write the **calibrate** function as defined above.

(b) **To hand in:** you are responsible for handing in the following:

1. A text file called **answers_hw2.txt** with your calibration results obtained from running your **calibrate** function on the target data we have provided.
2. An electronic copy of your MATLAB m-file for calibration called **calibrate.m**.

The code must be emailed to **cs223b@cs.stanford.edu** before midnight on the day the assignment is due.

2 Short answer (5 pts.)

To answer the following questions and to help understand the intuition behind edge detection and filters, you are encouraged to experiment with the following MATLAB functions: **edge** (type 'help edge' for instructions on how it works), **filter2**, **medfilt2**. MATLAB comes equipped with images to use for testing purposes. Some examples:

I1 = imread('rice.tif');
I2 = imread('eight.tif');
I3 = imread('pout.tif');

Add your answers to the **answers_hw2.txt** file you created earlier.

(a) **(1 pt.) Explain why median filtering performs well in images corrupted by impulse noise.**

(b) **(2 pts.) Explain the importance of hysteresis thresholding and non-maximal suppression in the Canny edge detection process. How do these two concepts influence the resulting edge image?**

(c) **(2 pts.) Is a ball better than a circle painted on a plane for determining a camera's aspect ratio? Provide a convincing argument (2 sentences) why your answer is the right one. How would you place the object in order to minimize the effects of lens distortion?**

(d) **To hand in:** you are responsible for handing in the following:

1. The text file called **answers_hw2.txt** with your answers to the previous questions.

This work must be emailed to **cs223b@cs.stanford.edu** by midnight on the day the assignment is due.