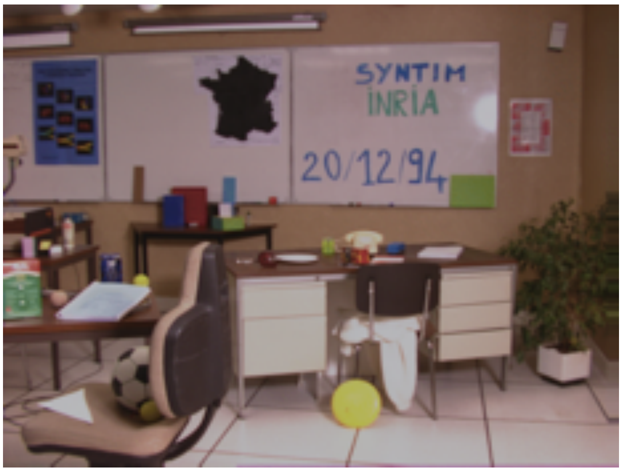
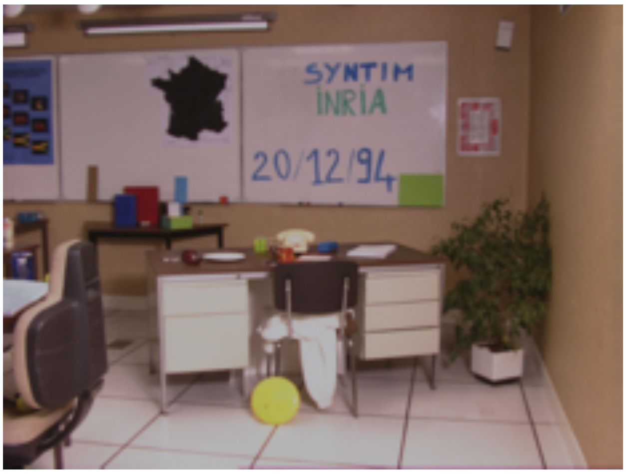
LAB 3: BINACULAR STEREO AND RECTIFICATION DUE 22TH NOVEMBER

Task 1 (30) Rectification





You are given a pair of images and their corresponding camera matrices P, in Sport_cam.mat. The are quite well aligned but not well enough for stereo matching. We want to distort the images, so the epipolar lines are paralell to the scanline and they are align between both images.

We will use the simple method of Fusiello et al. The paper, a tutorial, and Matlab code can be found here. http://www.diegm.uniud.it/fusiello/demo/rect/

For this exercise we are interested in the first paper: A compact algorithm for rectification of stereo pairs. You have an online tutorial here. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/FUSIELLO2/rectif_cvol.html.

The Result of this task is 2 homographies (3x3 matrices) that apply a perspective transformation to rectify the images.

NOTE

There is a detail not explained in the tutorial. You should align the 2 images so the vertical displacement is the same. To do so, once you calculate the Homographies, you use it to find the coordinates of the center of the image in the rectified image, and compute the displacement with respect to the original. Then, you calculate a new homography with the same process as before, but including the this displacement in the K. K1[0,2] = K1[0,2]+Dx1, K1[1,2] = K1[1,2]+Dy2 and similarly for the new K2.

You have the matlab code, so you can debug your solution, but try to implement it first. You can run this code "almost" directly in OCTAVE.

more in the next page…

Task 2 (30) Homography

Once an homography is computed for each image, you should apply it to each image. The result should be a rectify image as above. Your new image is bigger than the old one, so you need to compute the inverse of your homography, and then transform the corners, so you find the new dimensions. Then use some offsets to be able to account for the displacements.

If you have problems to apply the homography, you can use the provided rectified images for the following tasks.

Task 3 (20) Disparity Map

You have two options here: You can implement a simple stereo correspondence matching and compute a disparity map. Compute the cost for each pixel using a simple metric. You can of course implement more complicated things,

The second option is to use an implementation from OpenCV. Here an option.

http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_calib3d/py_depthmap/py_depthmap.html

You can check other options. Note that you might have to install OpenCV on your anaconda package.

Task 4 (20): Depth

Task 3 will should give you a disparity map. You can easily convert it to a depth map using the formula:

$$\text{disparity} = x - x' = \frac{Bf}{Z}$$

Where B is the baseline (distance between camera centres) and f is the focal length in pixels. Z is what you want to find. :) All this data is easily found from the camera matrices. You can save the result as a 3D model as in Lab2 or display the points using scatter 3D.

Extra: OpenCV stereo matching.

OpenCV provides several implementations of stereo matching, you can check them out. Here an option

http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_calib3d/py_depthmap/py_depthmap.html