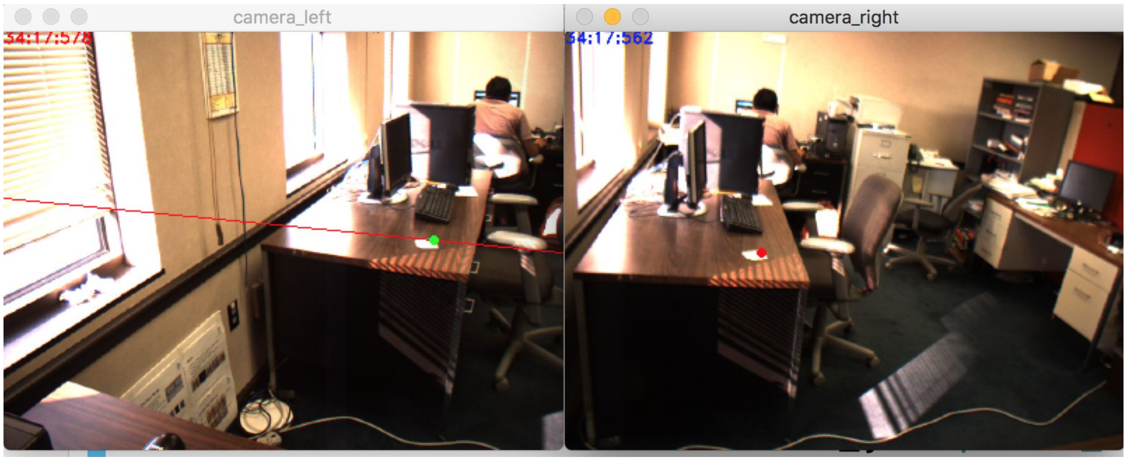
Two-view Geometry and Correspondence Matching

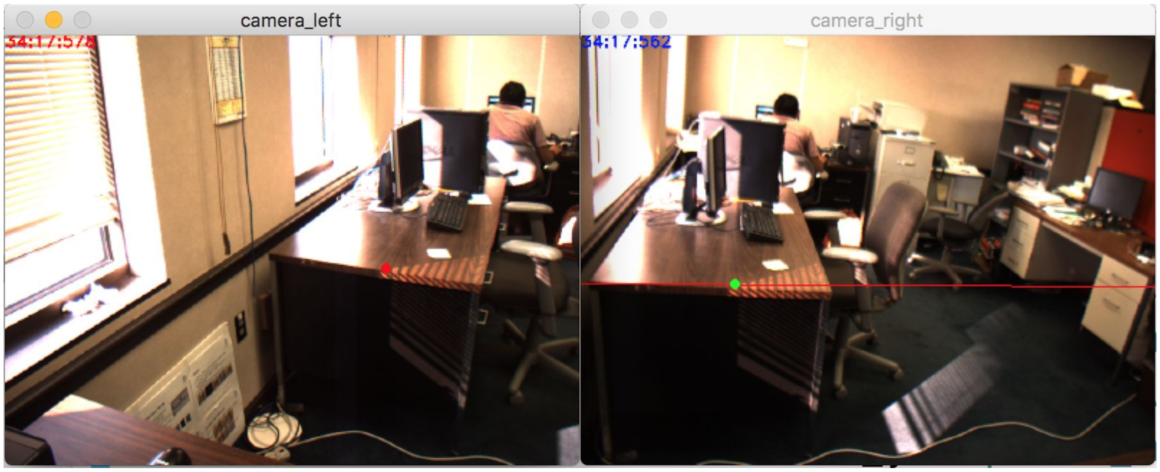
For this project, you will implement the epipolar geometry and normalized cross correlation (NCC) to find correspondences across two calibrated camera images. You are provided with the extrinsics between two cameras and their individual intrinsics. The goal is: given two captured images by the two cameras, your program is able to automatically find the correspondence between them using fundamental matrix transformation and NCC matching.

Input Data

(1) You are provided two image files:



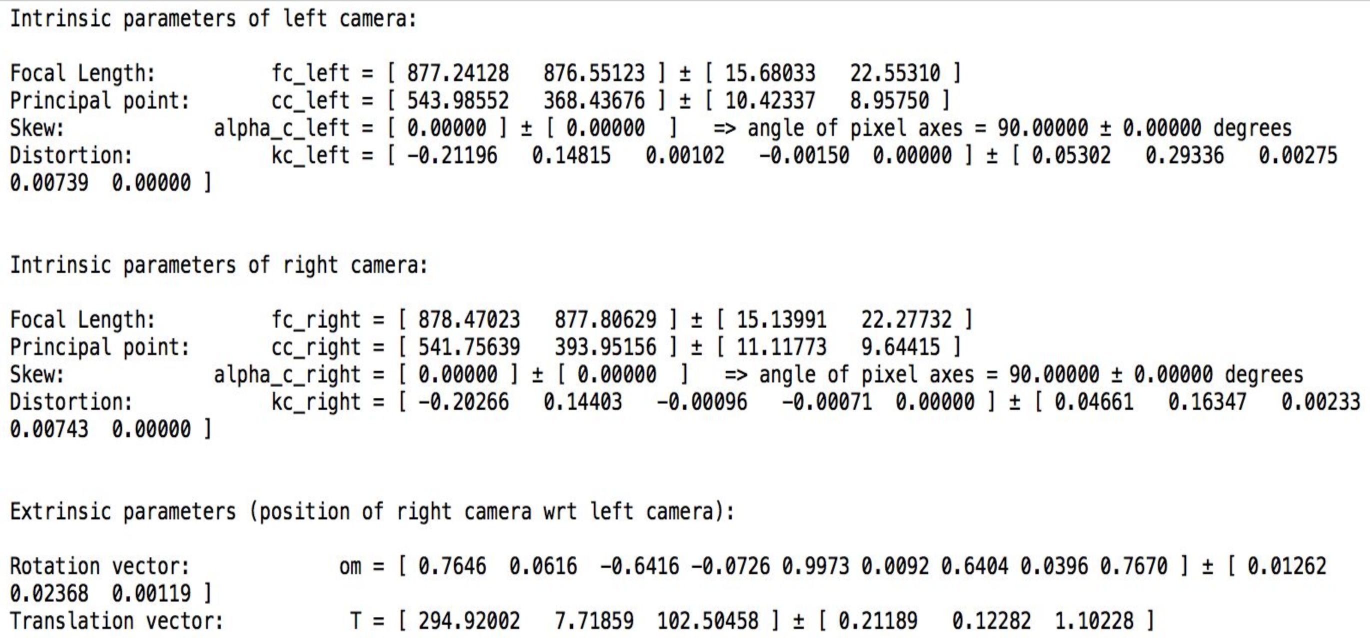
Example 1



Example 2

The two examples above demonstrate two images captured by the left and right cameras at the same time. By selecting one pixel on the right image (example 1) , an epipoloar line is automatically found from the left image and the correspondence is identified by NCC (indicated by the green circle). Example 2 shows a similar example but selecting a pixel from the left the image and found the correspondence from the right one.

(2) The calibration result file (processed by Matlab Toolbox):



```
Intrinsic parameters of left camera:

Focal Length:          fc_left = [ 877.24128   876.55123 ] ± [ 15.68033   22.55310 ]
Principal point:       cc_left = [ 543.98552   368.43676 ] ± [ 10.42337   8.95750 ]
Skew:             alpha_c_left = [ 0.00000 ] ± [ 0.00000  ]   => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:            kc_left = [ -0.21196   0.14815   0.00102   -0.00150  0.00000 ] ± [ 0.05302   0.29336   0.00275
0.00739  0.00000 ]


Intrinsic parameters of right camera:

Focal Length:          fc_right = [ 878.47023   877.80629 ] ± [ 15.13991   22.27732 ]
Principal point:       cc_right = [ 541.75639   393.95156 ] ± [ 11.11773   9.64415 ]
Skew:             alpha_c_right = [ 0.00000 ] ± [ 0.00000  ]   => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion:            kc_right = [ -0.20266   0.14403   -0.00096   -0.00071  0.00000 ] ± [ 0.04661   0.16347   0.00233
0.00743  0.00000 ]


Extrinsic parameters (position of right camera wrt left camera):

Rotation vector:             om = [ 0.7646  0.0616  -0.6416 -0.0726 0.9973 0.0092 0.6404 0.0396 0.7670 ] ± [ 0.01262
0.02368  0.00119 ]
Translation vector:           T = [ 294.92002   7.71859  102.50458 ] ± [ 0.21189   0.12282  1.10228 ]
```

This sample shows the two cameras' intrinsics and extrinsics:
fc_left - the left camera's focal length (fx and fy)
cc_left - the left camera's center offset (cx and cy)
Rotation vector - the relative rotation between the left and right cameras.
Translation vector - the relative position between the left and right cameras.

The data after the "±" symbol are the error deviation, you can simply ignore them. Also the "skew" and "distortion" are parameters about the lens, which are not necessary for the project.

What you should achieve?

(1) Successfully load and display two (left and right) color images at the same time. (10%)
(2) Successfully load the intrinsics and extrinsics file (.txt) into OpenCV matrix. (10%)
(3) Successfully use camera intrinsics and extrinsics to assemble a fundamental matrix. (25%)
(4) User selects a pixel on the left image by left mouse clicking. Your code should successfully render a corresponding epipolar line on the right image. This epipolar line should pass through the correspondence on the right image. (25%)
(5) Successfully implement NCC to find correspondence from the right image. (25%)

(6) Write up a “.pdf” document with several pairs of sample images showing the rendered epipolar line and correspondence results. (5%)