Computer Vision
Exercise 1 report

Amrollah Seifoddini

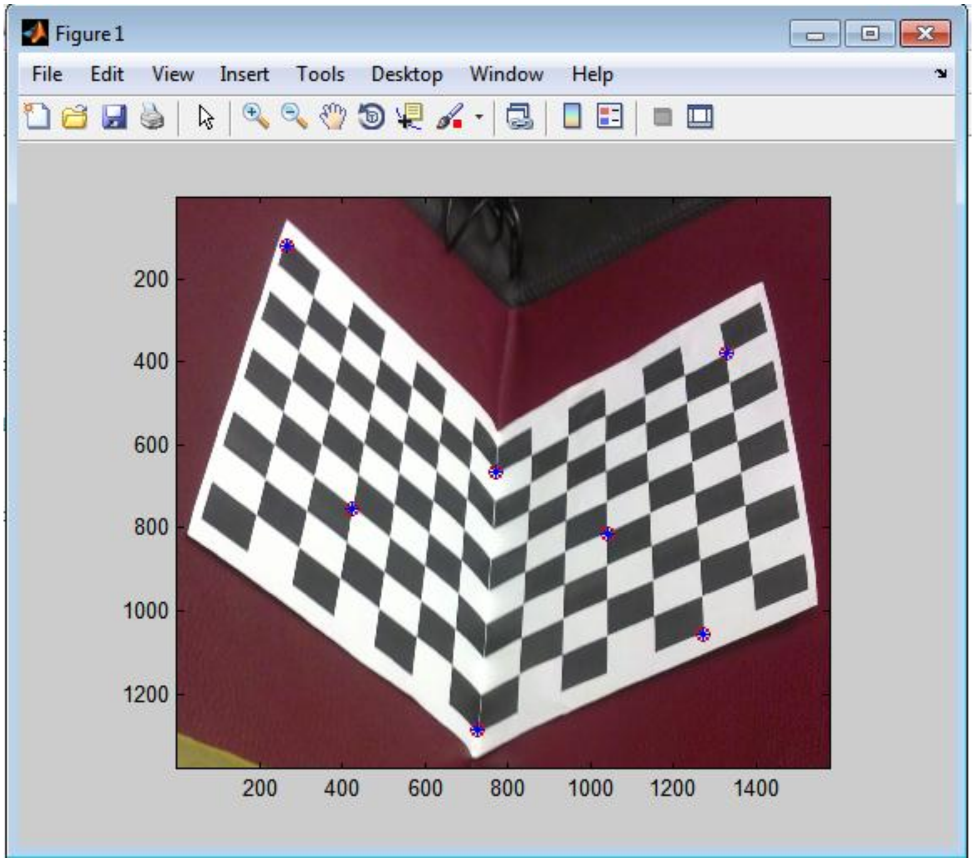
- The images are taken from Rabeeh Karimi, as all my efforts for connecting my mobile camera to the laptop failed.
- All the data, images are available in the attached zip file.
- The results of bouget's toolbox are available as .mat file in the attachment.

Part 1) DLT:

Error: 0.6012

```
K =     [-0.735377760744730   0.677657397322816     2.92529351734722e-05

        -0.677657387865848    -0.735377741663227    -0.000204296663905826

        -0.000116931188139268          -0.000170058690867749          0.999999978703569


R =     [118.740377067383      -22.4845353628441     24.0225945332980

        0          101.320478259727       58.7620350586501

        0          0          0.0346168671387168]


t =     [-1337.78558745219

        -432.297476585901

        0.721351038989325
```

For the implementation, I followed the exercise slides guide. First, normalization and then forming the matrix equation system is obvious. Afterwards, by using the SVD, I got V matrix and extract P out of it.



*Red circles are the original points and blue crosses are re-projected ones.

Part 2) Gold Standard Algorithm:

Error: 0.5984

K = [-0.735438300508732 0.677591695119998 2.91416626179427e-05

-0.677591685617086 -0.735438281397664 -0.000204542038899477

-0.000117164092588387 -0.000170174197765798 0.999999978656659]

R = [118.780249608207 -22.4823341934670 23.9958384487933

0 101.320457096799 58.7636048760055

0 0 0.0346335321671491]

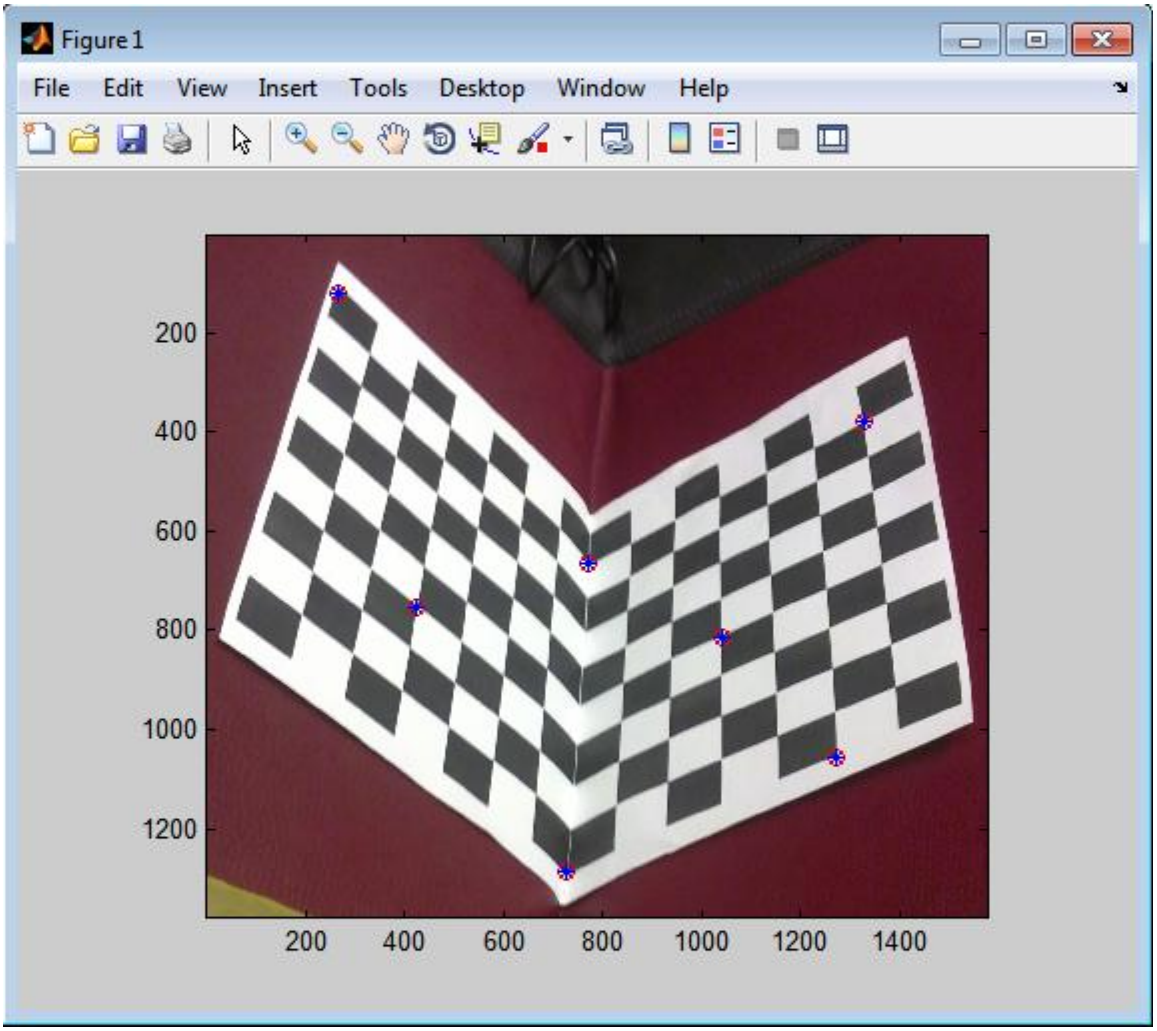
t = [-1337.76103053232

-432.403124683397

0.721005981538755]

For this part, I also followed the slides of the course and exercise. After normalization and initialization with DLT, I used an iterative loop for finding the best P.

As we can see, we have an improvement (although little!) compared to DLT.



Part 3) Bouget's toolbox:

The result are as follows for 14 images that I used.

Focal Length: fc = [3527.27973 3604.87629] ± [148.79011 167.75616]

Principal point: cc = [1307.38494 651.10609] ± [175.67058 181.79677]

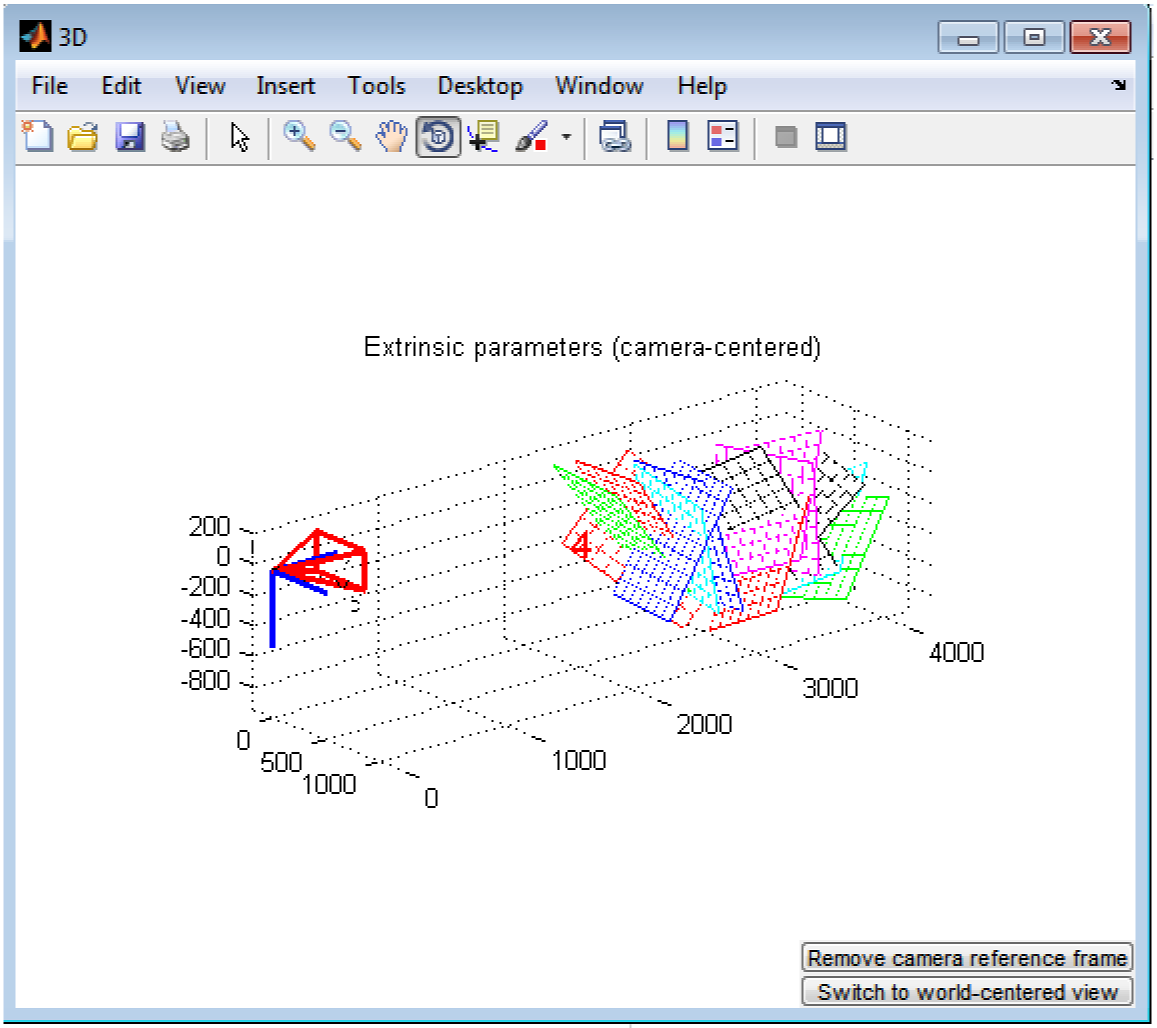
Skew: alpha_c = [0.00000] ± [0.00000] => angle of pixel axes = 90.00000 ± 0.00000 degrees

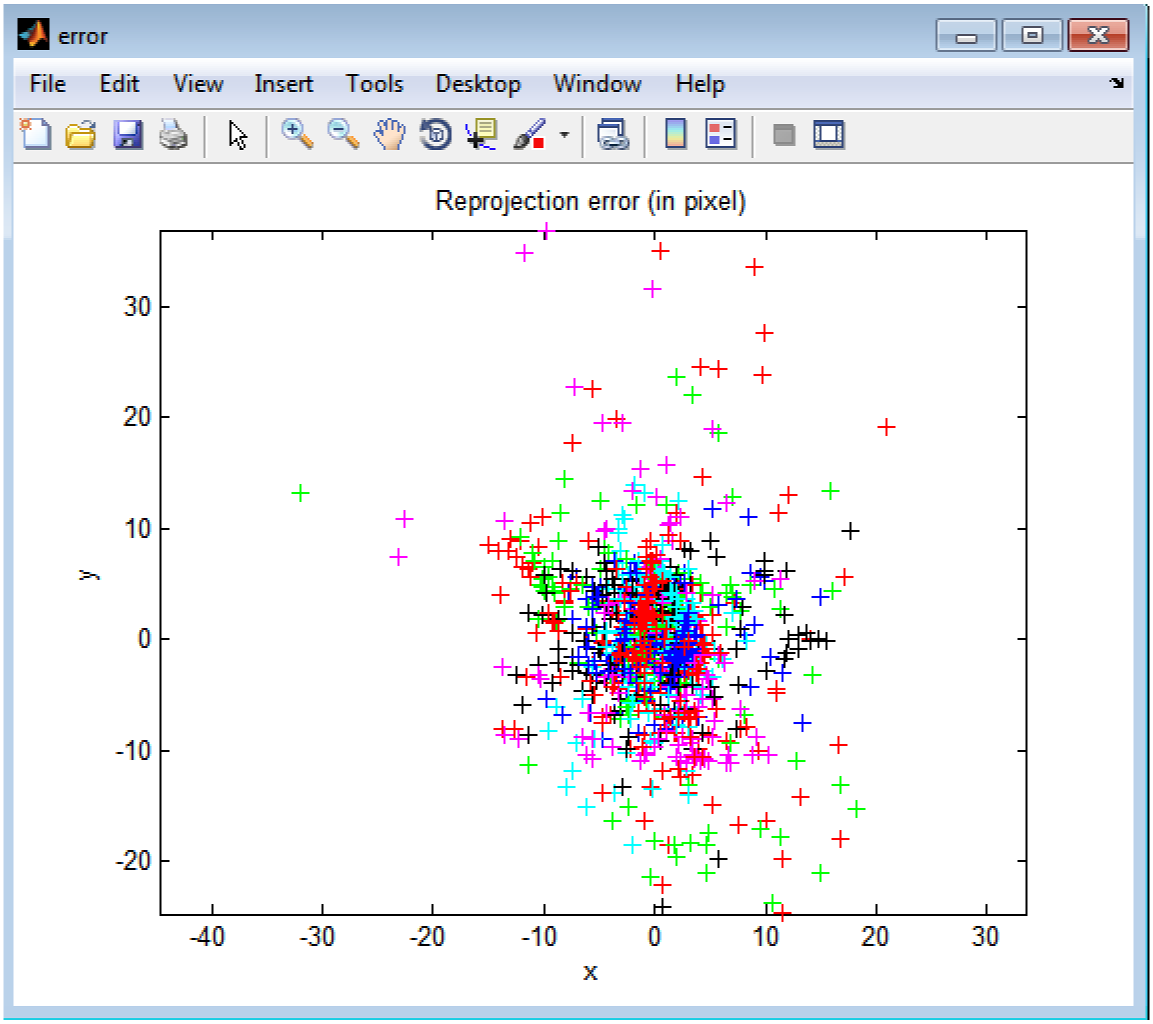
Distortion: kc = [0.08274 -0.56439 -0.01693 0.00463 0.00000] ± [0.17530 0.56595 0.01703 0.02165 0.00000]

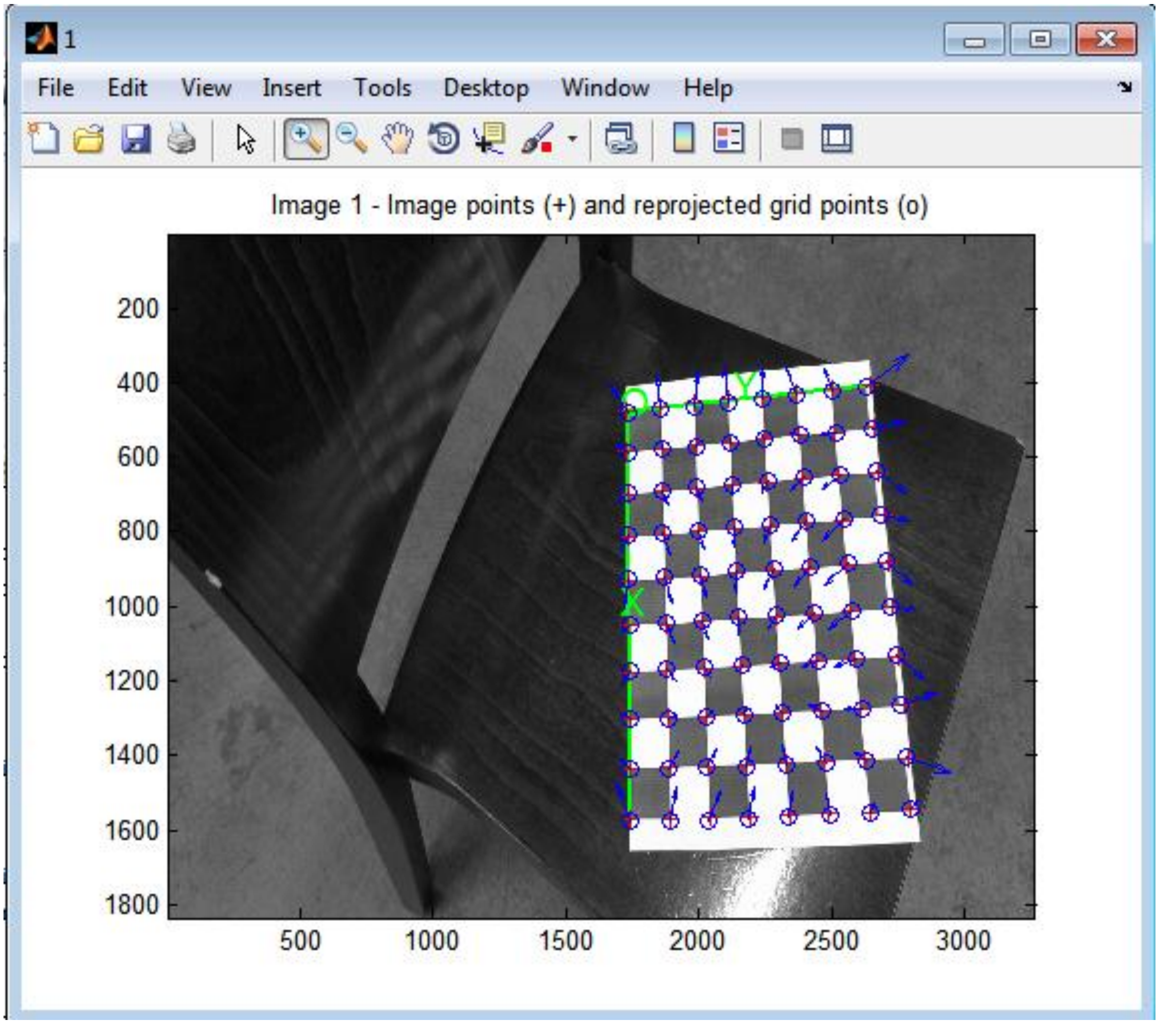
Pixel error: err = [5.28786 7.02424]

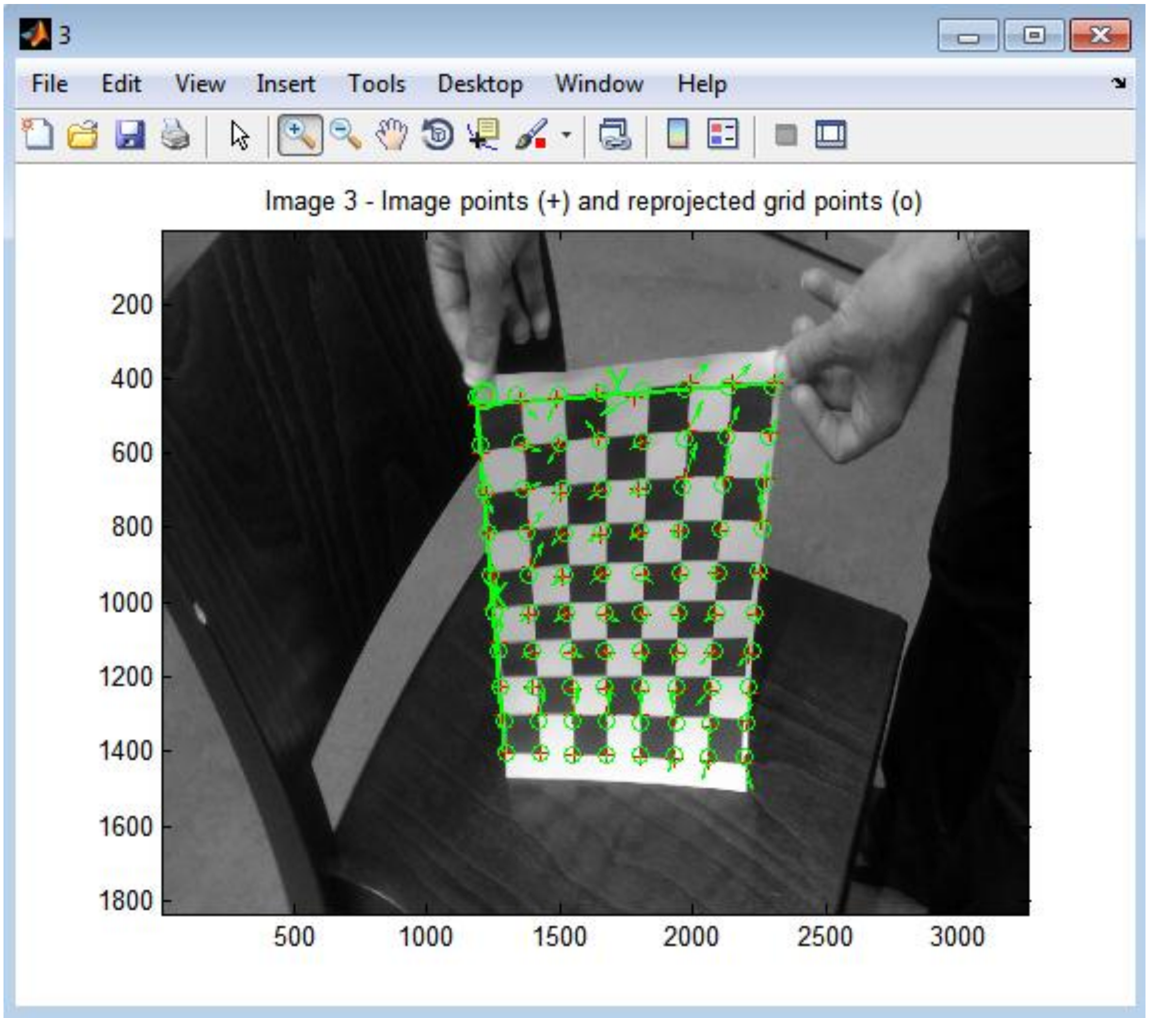
It can be seen that the error rate is much higher than DLT and Gold algorithms. I interpret this as a bad data issue. Because I used some steep angles and did not use the optional parameters that the toolbox offers, this is not a big surprise to see that much error. I mean the size of each square and window size and etc.

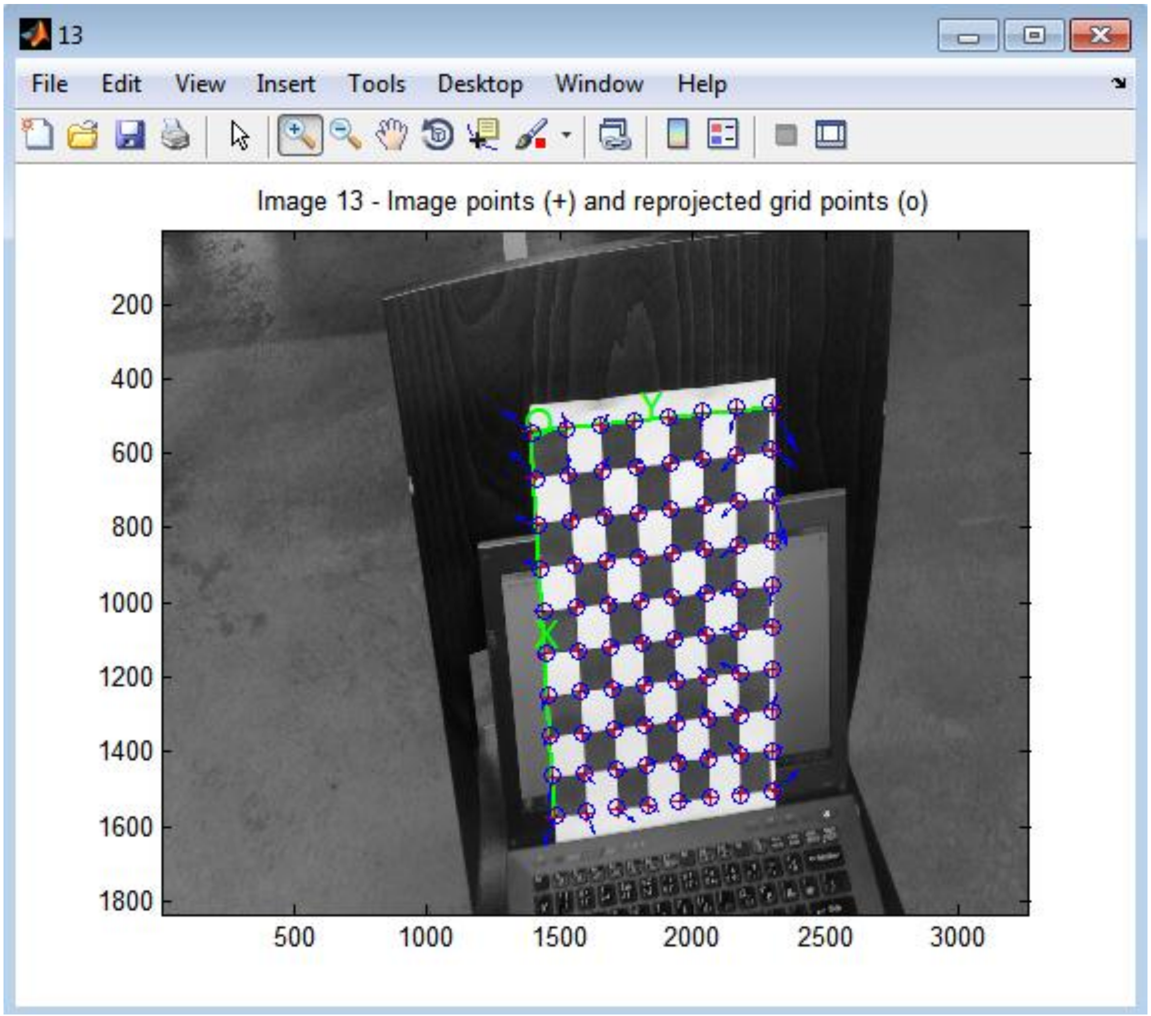
Some of the results are shown as pictures in the following.


3D
File Edit View Insert Tools Desktop Window Help
Extrinsic parameters (camera-centered)
200
0
-200
-400
-600
-800
0
500
1000
0
1000
2000
3000
4000
4
5
Remove camera reference frame
Switch to world-centered view


error
File Edit View Insert Tools Desktop Window Help
Reprojection error (in pixel)
30
20
10
0
-10
-20
y
-40 -30 -20 -10 0 10 20 30
x


1
File
Edit
View
Insert
Tools
Desktop
Window
Help
Image 1 - Image points (+) and reprojected grid points (o)
O
Y
X
200
400
600
800
1000
1200
1400
1600
1800
500
1000
1500
2000
2500
3000


3
File
Edit
View
Insert
Tools
Desktop
Window
Help
Image 3 - Image points (+) and reprojected grid points (o)
200
400
600
800
1000
1200
1400
1600
1800
500
1000
1500
2000
2500
3000
Y
X


13
File Edit View Insert Tools Desktop Window Help
Image 13 - Image points (+) and reprojected grid points (o)
200
400
600
800
1000
1200
1400
1600
1800
500
1000
1500
2000
2500
3000
O
X
Y