Experiment Report

Equipment:

Camera:

Canon PowerShot G7 X

Sensor size: 13.2mm x 8.8mm

Resolution: 20.20 Megapixels

Tripod:

AmazonBasics 60-Inch Lightweight Tripod

Practical Part 4a:

Step 1: Setup the experiment environment

I found a corner in living room with some decoration pictures and toys, which have relatively simple geometry (rectangles and cubes), choose two positons and took two pictures.

Step 2: Find some corresponding point pairs

In these two pictures, I marked 10 corresponding point pairs with its coordinate measured with Photoshop. Most of these point pairs are chosen from the corner point of a rectangle, or somewhere easy to recognize. With Photoshop and RAW format, I can easily zoom these pictures into every single pixel, which makes the measurement very accurate. But this means I must input the image points by my hand into the MATLAB code.

(P1 and p1 are a corresponding point pair represent the upper right corner point of the decoration picture's frame)

Step 3: Calculate the F-matrix

By saying "Calculate the F-matrix", in fact we are calculating the best solution of the F-matrix using least squares algorithm.

By plug my point pars into the 8-point algorithm (actually I have 10 point pairs)

$$[uu' \quad uv' \quad u \quad u'v \quad vv' \quad v \quad u' \quad v' \quad 1]\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

For convenience, make A = $[uu' \quad uv' \quad u \quad u'v \quad vv' \quad v \quad u' \quad v' \quad 1]$

Solving the homogeneous linear equations with MATLAB (`eig(A'*A)`) gives us a best solution F, F is the eigenvector of the first eigenvalue of $A^T A$.

My F-matrix:

```
F =

  -0.000000002041780   0.000000169475298  -0.000290982370414

  -0.000000021049994  -0.000000008400649  -0.001327304800374

   0.000019394519961   0.000812870400489   0.999998746227595
```

Setp 4: Calculate and plot epipolar line with the F-matrix

Now take P_1 (in the left image) and p_1 (in the right image) for calculation

$$L1: F * p_1 = \begin{bmatrix} -0.000000002041780 & 0.000000169475298 & -0.000290982370414 \\ -0.000000021049994 & -0.000000008400649 & -0.001327304800374 \\ 0.000019394519961 & 0.000812870400489 & 0.999998746227595 \end{bmatrix} * \begin{bmatrix} 2359 \\ 860 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} -0.000150050172660 \\ -0.001384186294479 \\ 1.744818963235570 \end{bmatrix}$$

$$\Longrightarrow L1:\ -0.000150050172660x - 0.001384186294479y + 1.744818963235570 = 0$$

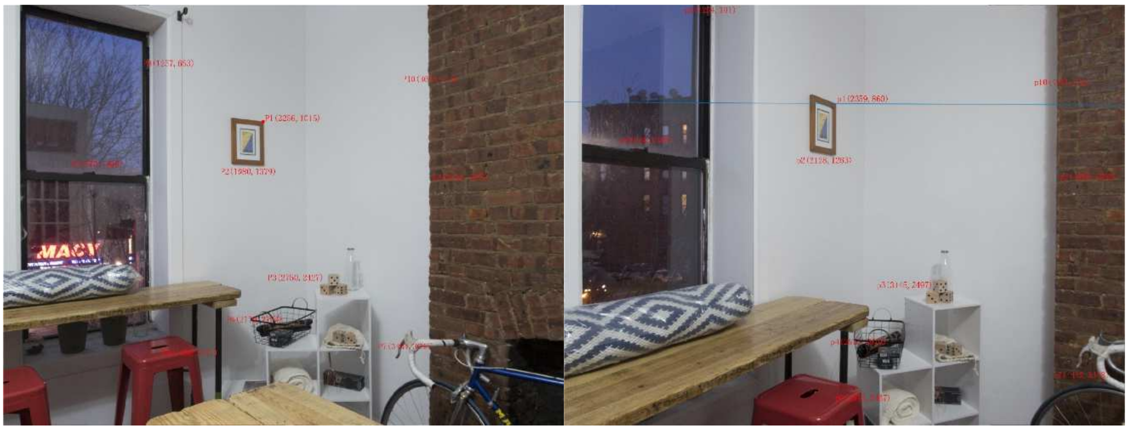
Plot in the image:

$$l1: {P_1}' * F = [2256 \quad 1015 \quad 1] * \begin{bmatrix} -0.000000002041780 & 0.000000169475298 & -0.000290982370414 \\ -0.000000021049994 & -0.000000008400649 & -0.001327304800374 \\ 0.000019394519961 & 0.000812870400489 & 0.999998746227595 \end{bmatrix}$$

$$= [-0.000006577479075 \quad 0.001186680013679 \quad -1.003671853807348]$$

$$\Longrightarrow l1:\ -0.000006577479075x + 0.001186680013679y - 1.003671853807348 = 0$$

Plot in the image:



Step 5: calculate epipole

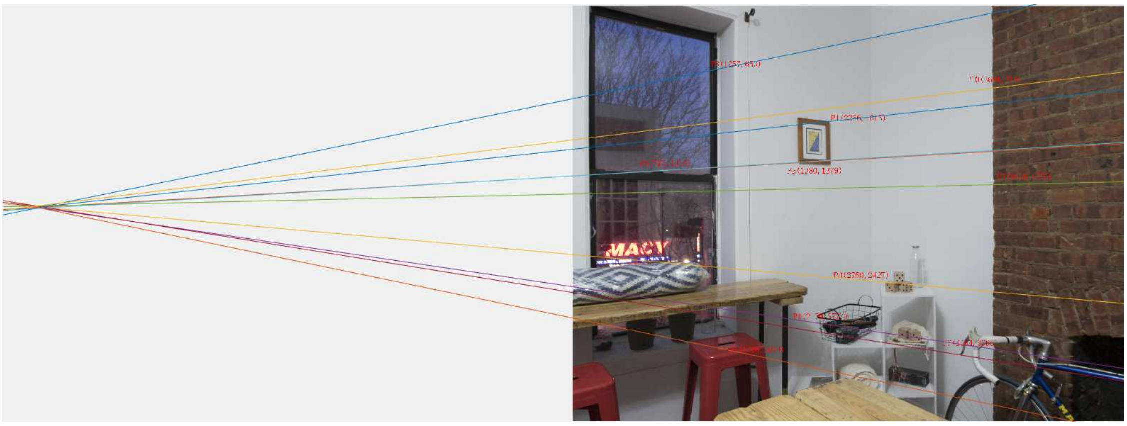
In left picture (shot by right camera)

Epipole is the best solution of F*x = 0, use same algorithm to fine a best solution `eig(F*F’)`

```
right epipole
   1.0e+03 *
  -4.707641851116959   1.785452392191962   0.001000000000000
```

x=-4708, y=1785

If we plot every epipolar line on the left image, we got:



Since my image size is 4864*3648, this epipole is quite feasible. This point is quite much the same positon where I put my left camera in.

In right picture (shot by left camera)

Epipole is the best solution of F*x = 0, use same algorithm to fine a best solution `eig(F'*F)`

```
left epipole
   1.0e+04 *
  -6.505026744348325   0.032184710627361   0.000100000000000
```

x=-65050, y=322

If we plot every epipolar line on the left image, we got:

This result maybe implies that

1) right camera's center is very close to the plane go through the left camera center and parallel to left camera's image plane, so all the epipolar lines are nearly parallel (but not strictly parallel).

2) right camera's center is behind the plane go through the left camera center and parallel to left camera's image plane, that's why epipole is on the left side of left camera.

Practical Part 4b:

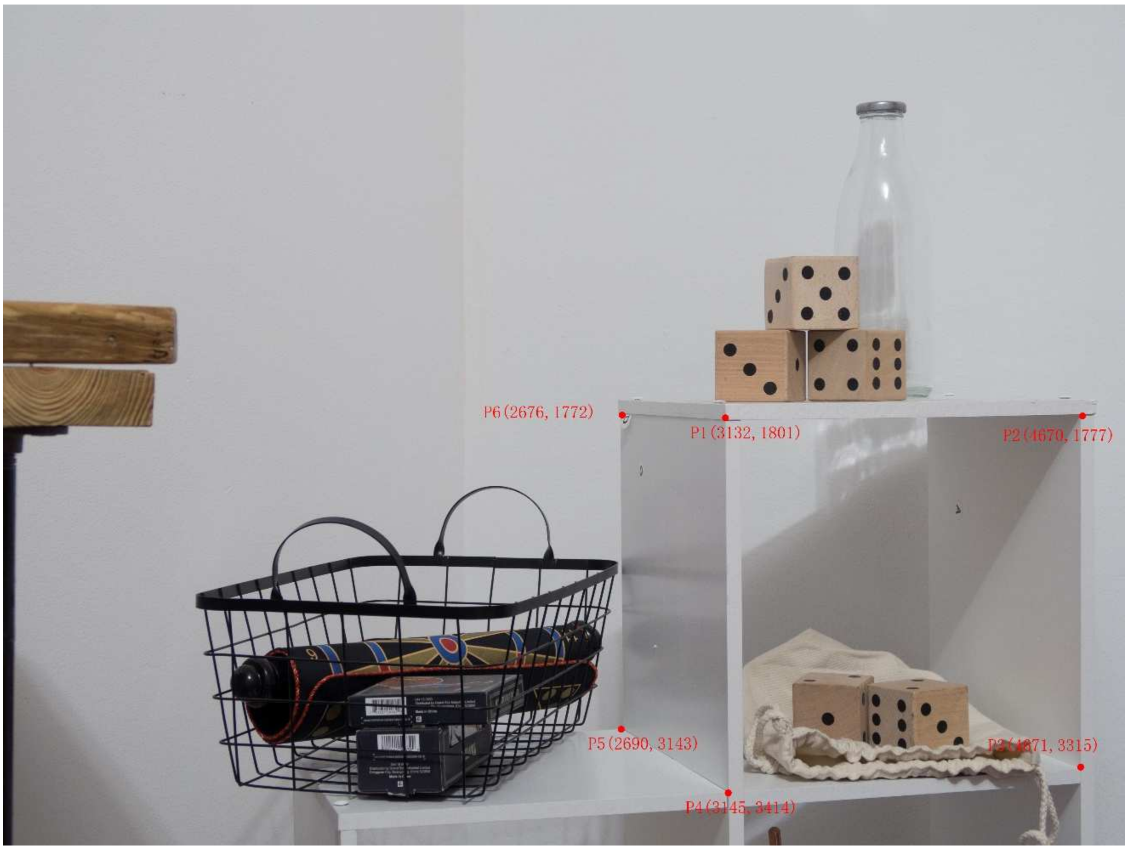
Step 1: Setup the experiment environment

I used a desk and a ruler to setup my parallel camera set: basically, slide my camera alone the metal ruler and record the baseline, and took two pictures before and after slide.

Step 2: Find corresponding point pairs

I took the white shelf as the target object, it has simple geometry and obvious key points: its corner.

I marked 6 corner in two pictures with photoshop and get their coordinate



Step 3: Calculate disparities

	Left image		Right image		Disparity pixel		Disparity mm	
	x	y	x	y	d_x	d_y	D_x	D_y
P1	3261	1787	3132	1801	-129	14	-0.3094	0.0336
P2	4794	1757	4670	1777	-124	20	-0.2974	0.0480
P3	4798	3295	4671	3315	-127	20	-0.3046	0.0480
P4	3279	3400	3145	3414	-134	14	-0.3213	0.0336
P5	2802	3130	2690	3143	-112	13	-0.2686	0.0312
P6	2785	1759	2676	1772	-109	13	-0.2614	0.0312

Step 4: Calculate scene points

With f=22mm, B=30mm, sensor density=417pixel/mm using

$$Z = -\frac{Bf}{d}, (d = \sqrt{d_x^2 + d_y^2})$$

$$\begin{cases} x = -f\dfrac{X}{Z} \\ y = -f\dfrac{Y}{Z} \end{cases} \Rightarrow \begin{cases} X = -\dfrac{Zx}{f} \\ Y = -\dfrac{Z\bar{y}}{f} \end{cases}, (\bar{y} = y_{left} + \frac{d_y}{2})$$

We can calculate scene point in camera coordinate

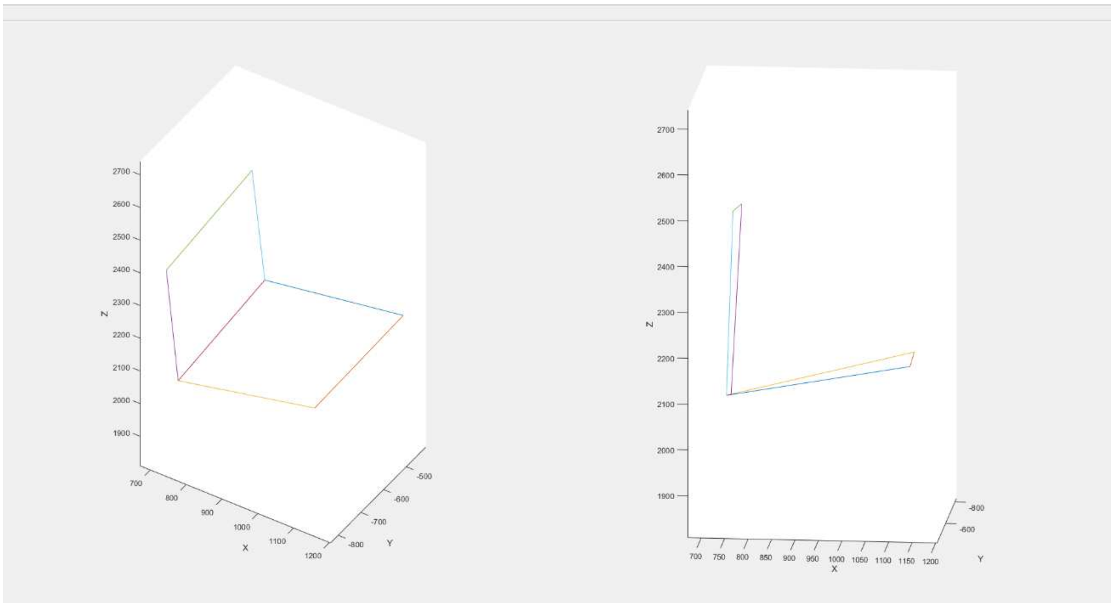
	Left camera(mm)		
	X	Y	Z
P1	753.95	414.77	-2121.03
P2	1145.04	422.05	-2191.20
P3	1119.59	771.20	-2140.70
P4	751.42	781.71	-2104.91
P5	745.53	834.53	-2440.93
P6	761.12	482.50	-2507.19

Theoretically, a parallel camera set should gain y=y', but with some noise and inaccurate (maybe a slightly rotation), every point has 10~20 pixel's disparity. Here I use (y+y')/2 as an estimate in calculation of Y.

And for the convenience, I didn't use the optical center as the origin of the coordinate system, instead, I choose the upper left corner in the plane parallel to the image plane and go through the optical center, the size of the plane is as large as the scene plane captured by my camera. Thus, I do not have to translate the coordinate but still got a same 3D reconstruction model.

Step 5: plot in 3D with MATLAB

I use plot3 to reconstruct these scene points, and chose different angle to observe the shape:



From some angle, the quality of reconstruction is pretty good, like the left image shows above. However, as the right image shows, in the above projection, the angle of the shelf is obviously not 90 degree. Maybe the inaccurate of parallel and some rotation is the reason.