1. **Estimating the fundamental matrix**

To estimating the fundamental matrix F we need to manually select correspondence points from each image. We have $P_i = (x_i, y_i, 1)$ is a pixel location in homogeneous coordinate in the image 1 and $P_i' = (x_i', y_i', 1)$ is a pixel location in homogeneous coordinate in the image 2.

   a. **Normalization**

First we need to normalize the data we have to find out two transform matrix T1(for P) and T2(for P'), that can make that all pixel correspondences have 0 mean and are at a distance of $\sqrt{2}$ from the center (0,0). The mathematical steps for normalization we have done that in projec 2 and 3 so I will skip those steps here

.

   b. **Find F**

Then as we know the F is the fundamental matrix we are looking for, and the relationship between point P_i from image 1,and point P_i' from image 2, and F is:

$$P_i'^T F P_i = 0$$

So we could obtain a equation:

$$x_i x_i' f_{11} + y_i x_i' f_{12} + x_i' f_{13} + x_i y_i' f_{21} + y_i y_i' f_{22} + y_i' f_{23} + x_i f_{31} + y_i f_{32} + f_{33} = 0$$

In matrix form:

$$\begin{bmatrix} x_i x_i' & y_i x_i' & x_i' & x_i y_i' & y_i y_i' & y_i' & x_i & y_i & 1 \end{bmatrix} f = 0$$

And if we apply all points we obtains:

$$\begin{bmatrix} x_1 x_1' & y_1 x_1' & x_1' & x_1 y_1' & y_1 y_1' & y_1' & x_1 & y_1 & 1 \\ x_2 x_2' & y_2 x_2' & x_2' & x_2 y_2' & y_2 y_2' & y_2' & x_2 & y_2 & 1 \\ x_3 x_3' & y_3 x_3' & x_3' & x_3 y_3' & y_3 y_3' & y_3' & x_3 & y_3 & 1 \\ \cdots & & & & & & & & \\ x_i x_i' & y_i x_i' & x_i' & x_i y_i' & y_i y_i' & y_i' & x_i & y_i & 1 \end{bmatrix} f = 0$$

We simplify this in the form $Af = 0$, then we solved this use SVD and the solution can be obtained corresponding to the smallest singular value of the SVD result. From the smallest singular value we can extract $\hat{F}$.

The next step we want to rank the $\hat{F}$ to 2. Because the $\hat{F}$ we obtained has a rank of 3 and it could not satisfy the equation $P_i'^T F P_i = 0$. And so is rank 1. So we do $F = UDV^T$ and set the smallest singular value in D equal to 0.

After that, we obtain a F will satisfy our requirements. So we want to de-normalize the it by the equation

$$F = T_2^T F T_1$$

Then we obtain the F we want.

2. **Find camera matrix P**
    a. **Find e1 and e2**

Epipoles e1(for image 1) and e2(for image 2) are the right and left null vectors of fundamental matrix F. We can decompose to obtain them.

$$F = UDV$$

And we have

$$e1 = \begin{bmatrix} V(3,1) \\ V(3,2) \\ V(3,3) \\ V(3,4) \end{bmatrix} e2 = \begin{bmatrix} U(3,1) \\ U(3,2) \\ U(3,3) \\ U(3,4) \end{bmatrix}$$

    b. **Find P1 and P2**

Project P1(for image 1) and P2(for image 2) are the project matrix that we want to use to find 3D coordinate for our desired points sets. We initial P1 as:

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

And we can obtain P2 using:

$$P_2 = \begin{bmatrix} e_2 F | e_2 \end{bmatrix}$$

3. **Reproject the points back to 3D**

For each image we have a Project matrix. We represent as:

$$P = \begin{bmatrix} P.row1 \\ P.row2 \\ P.row3 \end{bmatrix}$$

For each point (x_i^1, y_i^1) and (x_i^2, y_i^2), we form the matrix:

$$A = \begin{bmatrix} x_i^1 P_1.row3 - P_1.row1 \\ y_i^1 P_1.row3 - P_1.row2 \\ x_i^2 P_2.row3 - P_2.row1 \\ y_i^2 P_2.row3 - P_2.row2 \end{bmatrix}$$

Then we do a SVD on A, and the smallest singular value of the result V is the 3D coordinate of the 2 correlated points on the two images.

4. **Find plane of infinity**

For easier computation, we first find vanishing points in 2D images. To construct a plane of infinity, we need 3 vanishing points. Each vanishing points needs two parallel lines to construct and each line is the cross product of 2 points on the line.

We first pick 4 points A, B, C, D on image 1 and A' and B' on image 2. A and A' is the same point on the 3D correspondence. Line AB and CD are parallel line on the real world

So we can have:

$$Line_{AB} = A \times B$$
$$Line_{CD} = C \times D$$
$$VanishingPoint_{image1} = Line_{AB} \times Line_{CD}$$
$$VanishingPoint_{image2} = (F * VanishingPoint_{image1}) \times Line_{A'B'}$$

After obtaining two vanishing points from each image, we do the same trick in section 3 to obtain a A matrix. And the two vanishing points' 3D corresponding vanishing point can be found by find null vector of the A matrix we constructed.

We will find 3 vanishing points for both wall and floor(total of 6). We can construct a matrix, each row of this matrix is value of one vanishing point.

$$\begin{bmatrix} VanishPoint1 \\ VanishPoint2 \\ VanishPoint3 \end{bmatrix}$$

The plane of infinity can be found by finding the Null matrix of this matrix. It should looks like this:

$$PlaneOfInfinity = \begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix}$$

5. **Remove Affine**

The Affine matrix can be represented as:

$$H = [I|PlaneOfInfinity]$$

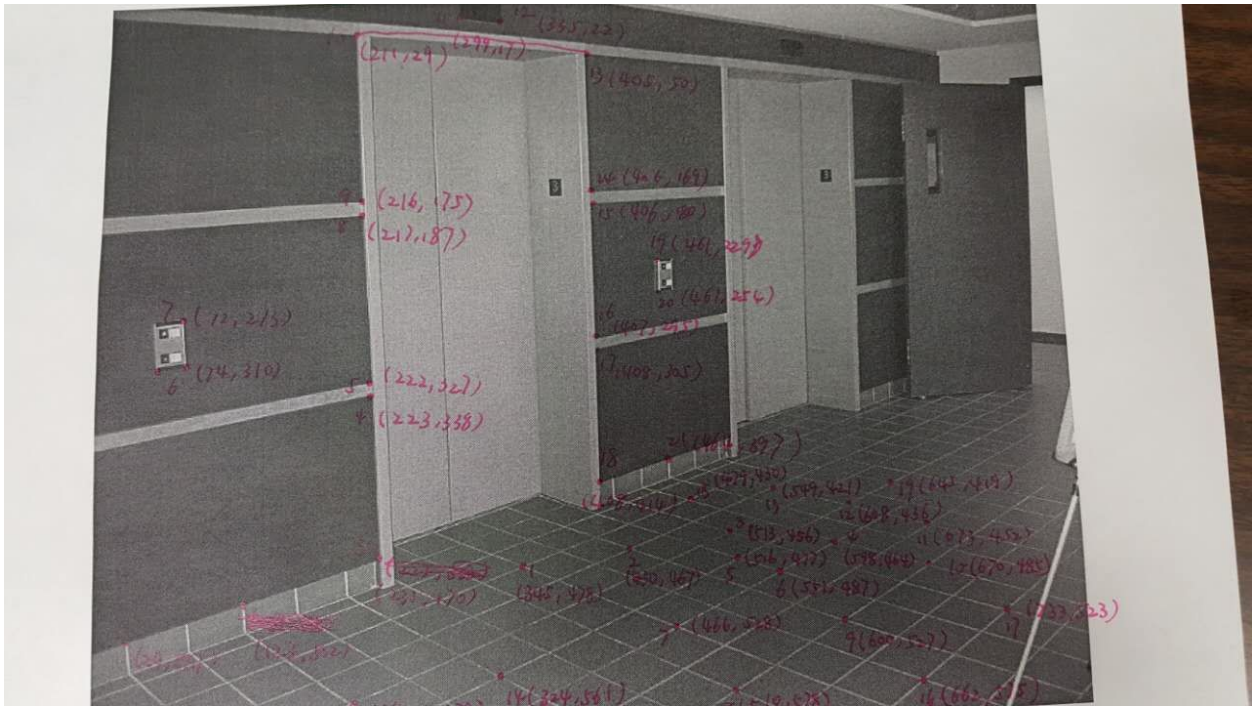
or

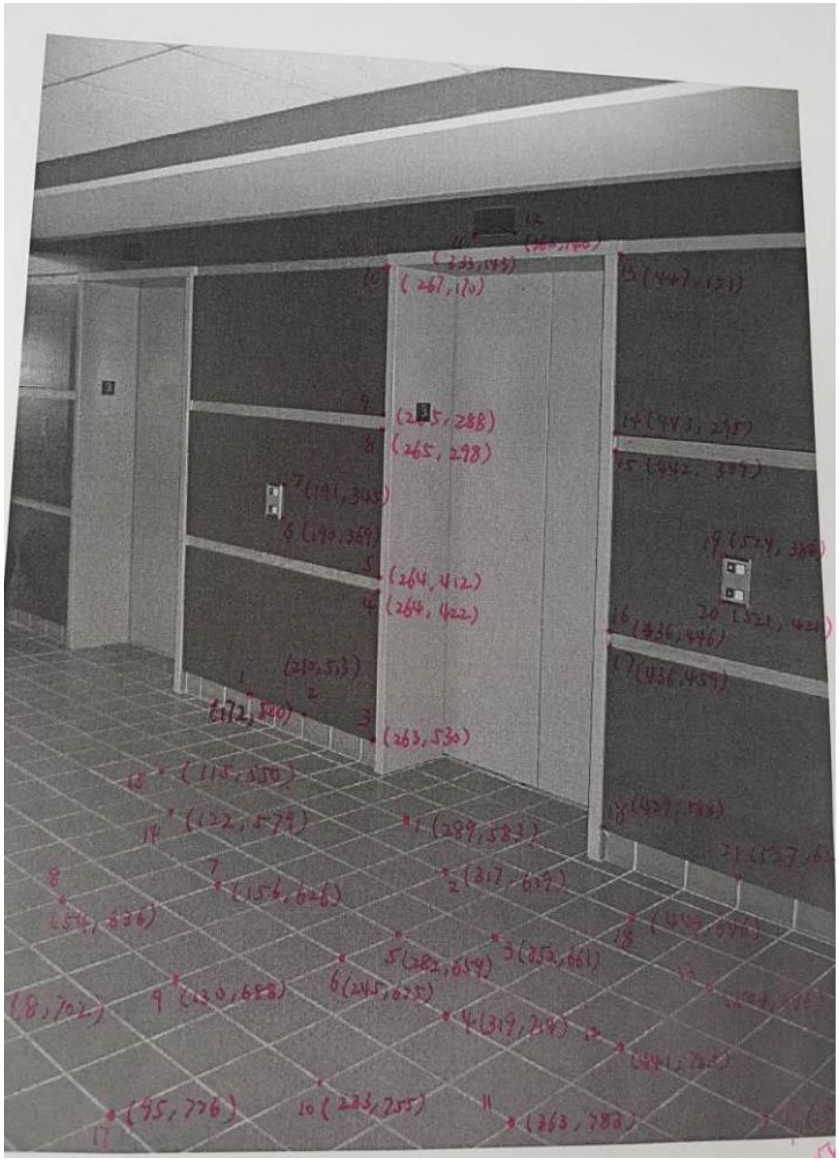
$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ A & B & C & D \end{bmatrix}$$

We can apply each points of 3D correspondence to its Homography to remove affine.

Result:

I select 40 points from both image, 21 for wall points and 19 for floor points. The picture below marked the points on the image.





Here is the F we found:

$$F = \begin{bmatrix} -1.7699668266554 86e-06 & 5.831095730050382e-06 & -0.0016295059228110 61 \\ 3.9742670666784 28e-06 & 8.322182576621457e-07 & 0.0028925244490302 41 \\ 0.0008638037204245528 & -0.006566185689964552 & 1 \end{bmatrix}$$

The Determinant of the F = -6.46235e-27, which is closed to 0, so the F we found is valid;
Based on the F, we obtained P1, and P2 for the both images, P1 is what we set up:

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

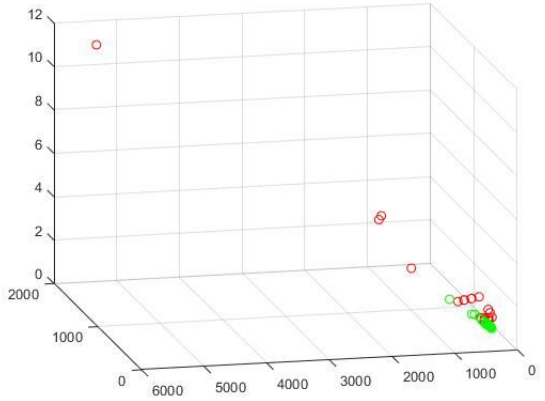
$$P_2 = \begin{bmatrix} -0.00020600 & 0.00156596 & -0.23848635 & -0.97114481 \\ 0.00083888 & -0.00637672 & -0.23848893 & 0.97114626 \\ -0.00319768 & -0.00000428 & 0.00000058 & -0.00089265 \end{bmatrix}$$

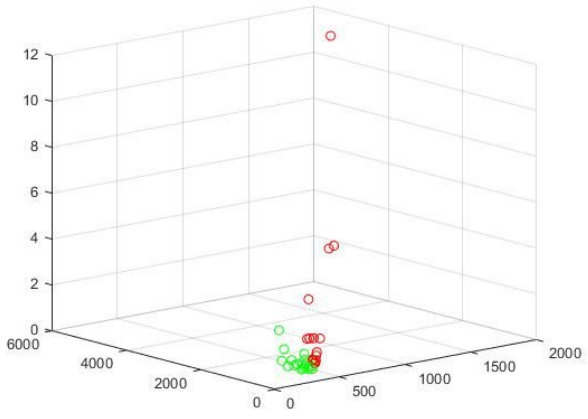
And here is the points we obtained for each pairs of points from the image:

{172,500}	{29 ,531}	1944.8372, 5359.3774, 11.014791;
{210,513}	{123,502}	618.11383, 1261.9188, 2.7209847
{263,530}	{230,470}	432.51367, 646.36664, 1.4623066
{264,422}	{223,338}	425.19635, 556.33679, 1.5280368
{264,412}	{222,327}	425.09802, 549.55017, 1.5363357
{190,369}	{74 ,310}	869.53339, 1602.9799, 4.5176363
{191,345}	{72 ,273}	893.09412, 1548.8268, 4.6372561
{265,298}	{217,187}	422.52228, 440.28458, 1.5835121
{265,288}	{216,175}	423.6134, 431.60727, 1.5907284
{267,170}	{211,29 }	435.65079, 306.64905, 1.6206239
{333,143}	{299,17 }	381.18845, 200.52794, 1.1286753
{365,140}	{335,22 }	369.50842, 177.9821, 0.99788427
{447,151}	{405,50 }	360.15833, 151.5661, 0.79790246
{443,295}	{406,169}	356.70291, 209.64932, 0.80007952
{442,309}	{406,180}	356.87491, 214.93036, 0.7997964
{436,446}	{407,295}	364.89792, 267.04825, 0.78483158
{436,459}	{408,305}	365.85458, 270.76593, 0.7809056
{429,588}	{408,414}	381.06842, 318.35352, 0.75793886
{524,386}	{461,229}	361.95703, 200.71906, 0.6696564
{521,421}	{461,254}	363.74799, 210.85121, 0.66690427
{527,621}	{464,397}	386.78815, 262.05106, 0.62543696
{289,583}	{345,478}	351.95529, 437.09778, 0.99247831

{317,619}	{430,467}	328.72122, 343.28775, 0.79188967
{352,661}	{513,456}	319.69516, 279.12476, 0.64975125
{319,714}	{598,464}	301.1351, 243.25177, 0.56477004
{282,659}	{516,477}	294.52957, 295.23999, 0.68637639
{245,625}	{551,487}	256.5769, 291.1127, 0.69581175
{156,626}	{466,528}	237.02458, 374.8367, 0.86760747
{54 ,636}	{510,578}	167.28897, 388.71851, 0.90526974
{130,688}	{600,527}	233.35031, 290.27856, 0.67398447
{233,755}	{670,485}	283.45261, 226.69868, 0.51902229
{363,783}	{673,452}	312.80148, 204.06496, 0.46801385
{441,735}	{608,436}	329.92554, 218.07693, 0.5060454
{504,696}	{549,421}	354.65781, 231.60783, 0.53921252
{122,579}	{324,561}	234.34903, 565.12091, 1.2639829
{115,550}	{208,579}	285.73615, 882.45715, 1.8941851
{8 ,702}	{662,575}	188.35791, 295.53241, 0.68339622
{95 ,776}	{733,523}	270.21857, 220.78668, 0.48764944
{449,646}	{408,414}	404.41678, 311.20911, 0.72161376
{544,785}	{645,419}	346.48834, 191.13576, 0.43966958

We plot all the point into Matlab, Green is floor as red is wall points. All the point are crowded in a line, however, the boundary is clear for floor and wall points.





The next is to find the plane of infinity of each plane.
For the wall plane we found a plane of infinity:

$$POI_{wall} = \begin{bmatrix} -0.0018 \\ 0.0002 \\ -1.0000 \\ 0.0000 \end{bmatrix}$$

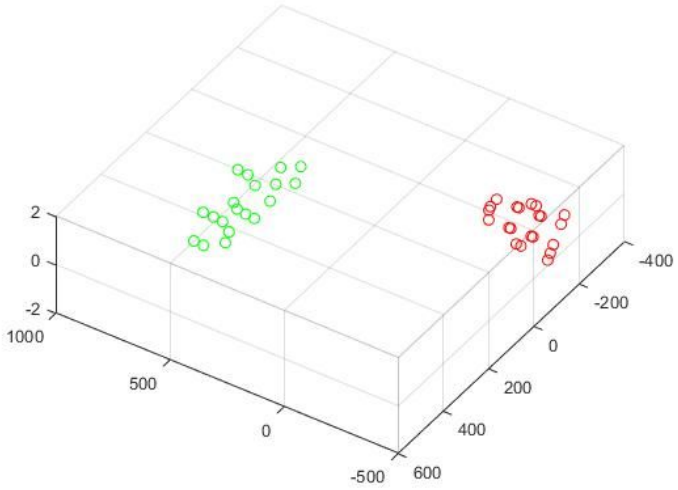
For the floor:

$$POI_{floor} = \begin{bmatrix} -0.0002 \\ 0.0034 \\ -1.0000 \\ 0.0000 \end{bmatrix}$$

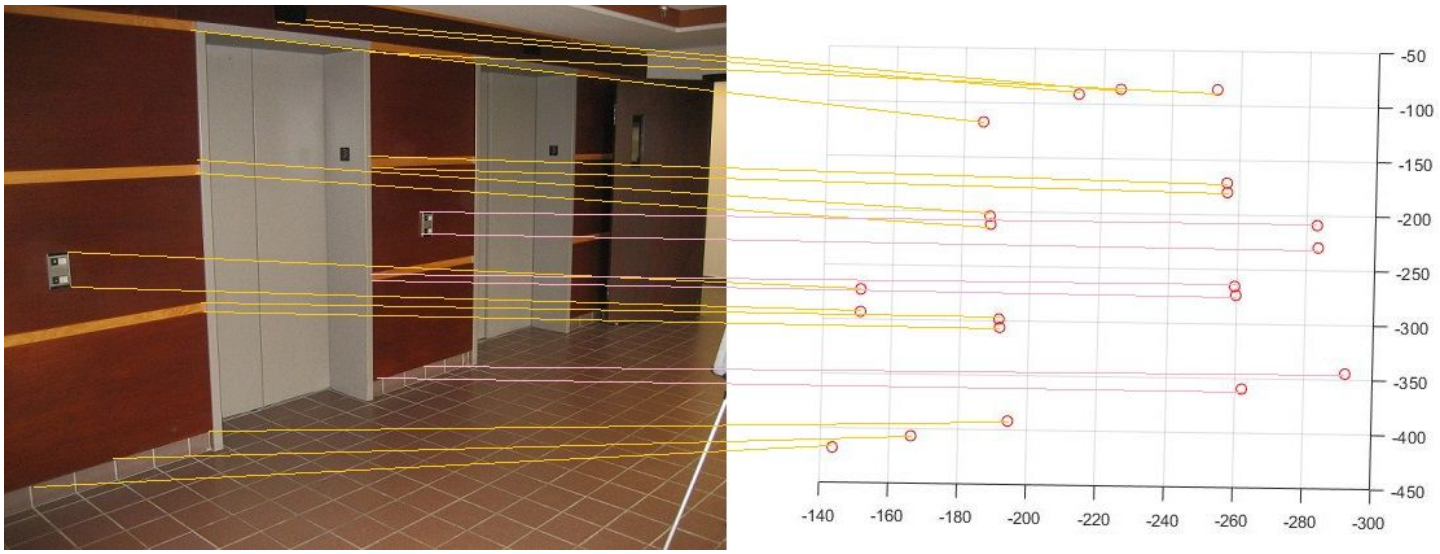
Then we obtain two homography for each plane transformation:

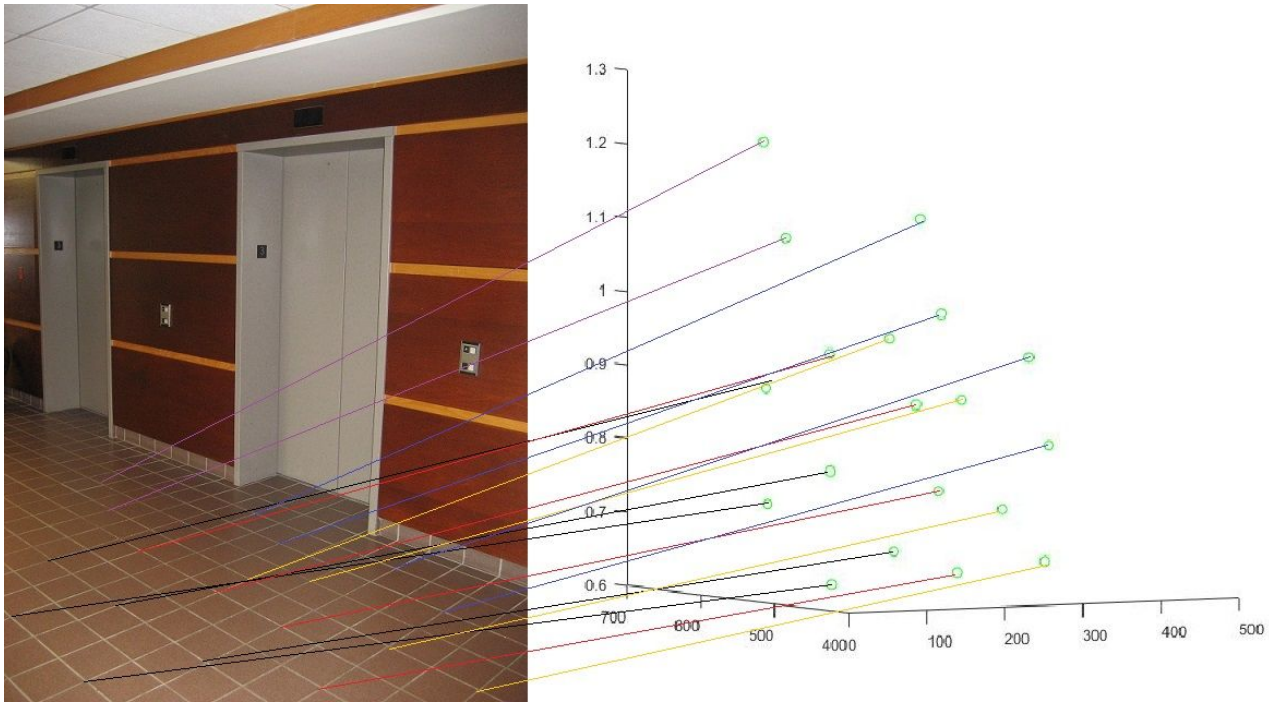
$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -0.0018 & 0.0002 & -1 & 0 \end{bmatrix} H_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -0.0002 & 0.0034 & -1 & 0 \end{bmatrix}$$

We apply point 1 to 21 to H1 as they belong to wall, and points 22 to 40 to H2 as they belong to floor. And we project them into the same scatter to see the result.

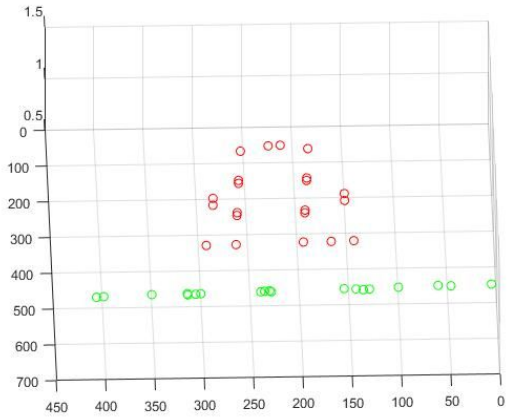


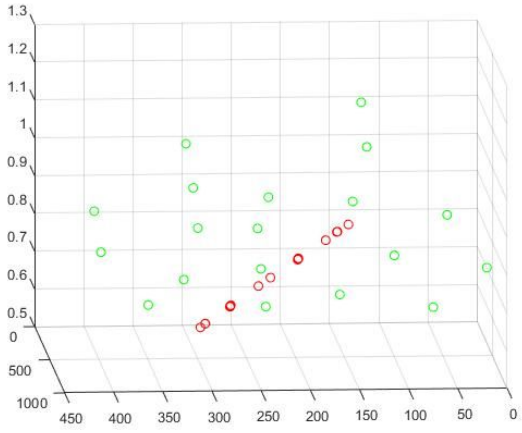
The floor and wall has been separate clearly and to validate the result, let's first take a look at each individual plane.(I rotated the image to match the image scenario, so the line looks not very parallel in this two image, we will have a ratificated look after those.)



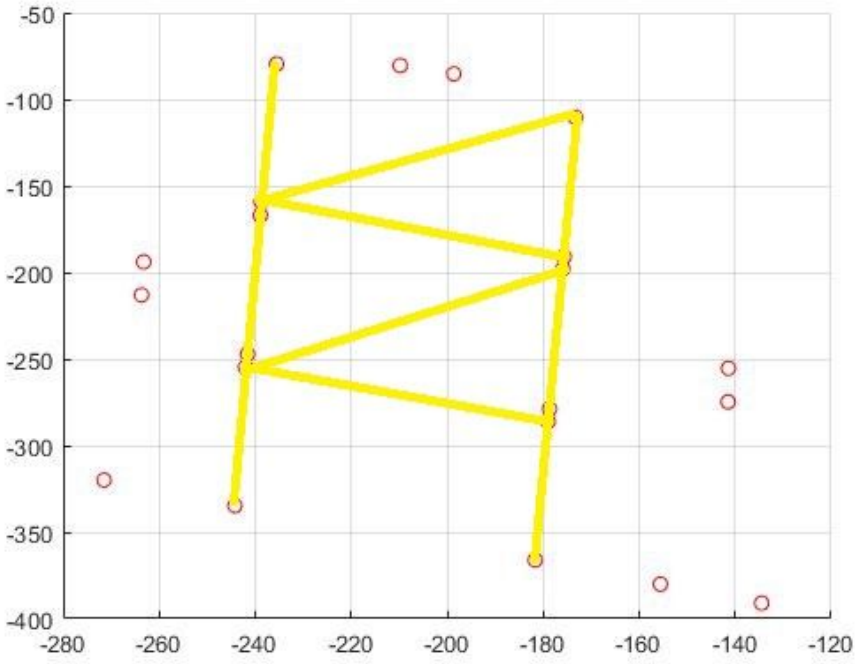


And I project two plane closer to see if they have a good angle within two plane:





And their parallel lines:



It obviously that the two plane are in a orthogonal angle.(There is little error when put two plane closer.)
Here is the new set of our 3D coordinates.

```
 Columns 1 through 8
-143.7070 -166.4031 -194.3734 -191.4694 -191.1619 -151.0158 -150.9572 -188.1963
-417.7532 -406.4994 -391.7034 -306.0608 -298.3283 -293.2884 -272.6711 -211.6319
  -0.8355   -0.7924   -0.7391   -0.7253   -0.7241   -0.7948   -0.7904   -0.7102
   1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
 Columns 9 through 16
-187.9003 -185.4070 -212.8927 -225.0174 -253.0582 -256.4706 -256.6189 -259.3272
-204.2084 -118.0494  -91.4224  -86.3081  -85.4851 -170.7874 -179.4009 -265.2751
  -0.7091   -0.6944   -0.6393   -0.6165   -0.5661   -0.5789   -0.5806   -0.5948
   1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
 Columns 17 through 24
-259.7727 -262.0246 -282.9290 -283.3181 -291.9186  315.5441  307.0026  301.2854
-273.4764 -359.1391 -208.4171 -228.9383 -343.9881  636.5478  599.4783  565.7660
  -0.5958   -0.6108   -0.5399   -0.5438   -0.5539    1.0918    0.9685    0.8559
   1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
 Columns 25 through 32
 235.9579  240.4615  230.2132  144.2093   47.7176  100.4215  154.9608  230.1656
 528.1309  561.9291  587.2816  578.6851  562.0061  531.4608  502.1257  496.4723
   0.7397    0.8527    0.9396    0.9244    0.8837    0.7725    0.6651    0.6341
   1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
 Columns 33 through 40
 314.2872  399.3975  131.3222  138.1318    5.8774   59.5140  407.2954  349.8326
 523.8118  551.5491  623.2412  660.6295  515.7633  486.1374  585.9951  504.8134
   0.7127    0.7925    1.0764    1.2011    0.7347    0.6265    0.9071    0.6431
   1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000
```

Discussion:

The result is perfect if we only look at individual plane result, parallel lines are parallel, and all points are at their correct position. Two plane would form an orthogonal angle

However, when we merge them together, however the edge not connect in a correct angle(we suppose floor's edge and wall's edge form a 0 angle, however the 3D object we obtained formed a 10-20 degree angle.), I will assume this is because of the points we selected from two image doesn't very accuracy.

Also the calculation and SVD can not be very accurate.