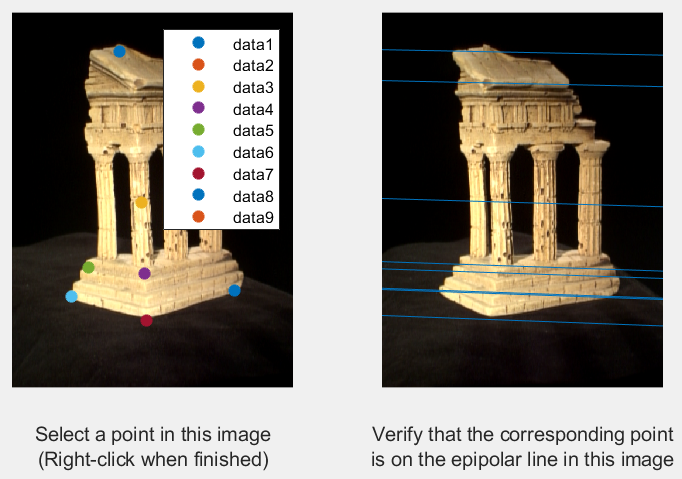
3.1.1:

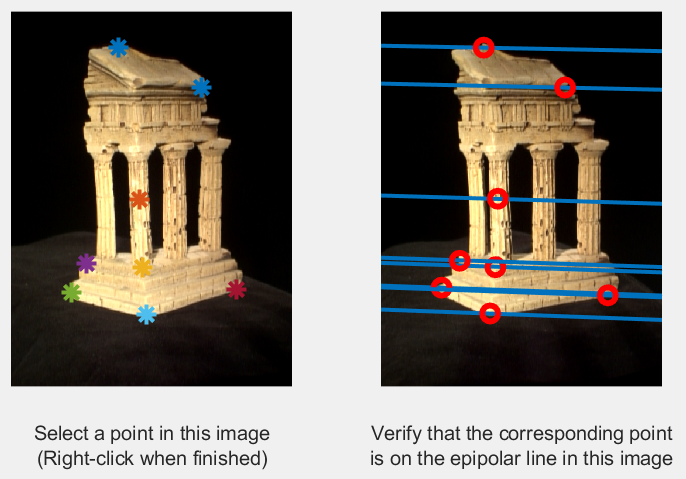
F = [1.75183168875261e-09, -1.86674315689630e-08, -8.52016381160499e-06;

-6.45671395631849e-08, -4.02137867037848e-10, 0.000495676907212069;

1.66353907424753e-05, -0.000476097927042134, -0.00205693230902430;]



3.1.2:



I used Euclidean distance for similarity metric. This matching algorithm might fail when pictures have repeated patterns, because repeated patterns have similar Euclidean distance. It also might fail when pictures have areas with no pattern but pure color, because points in these areas also have similar Euclidean distance.

3.1.3:

E = [0.00404956244132062, -0.0433080372767750, -0.0191554874996294;

-0.149794366553689, -0.000936326071206588, 0.726416434975663;

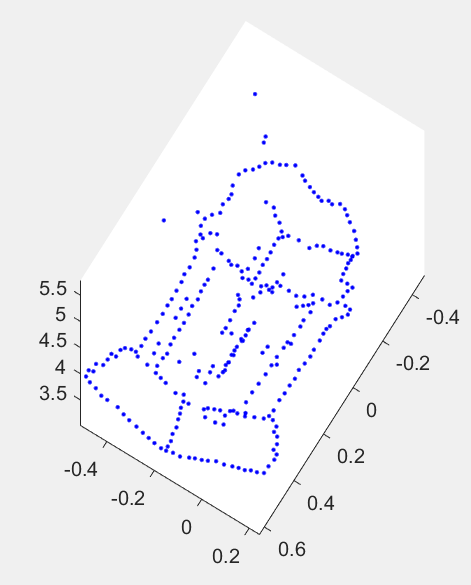
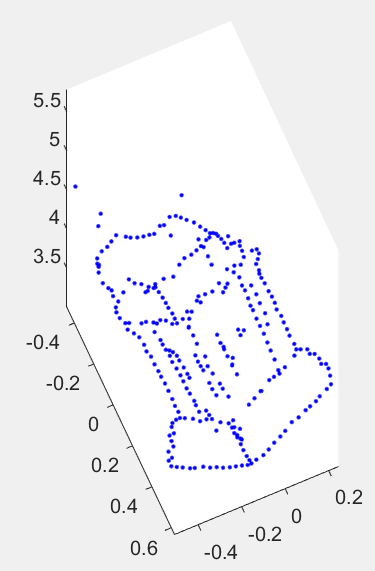
0.00186296855297823, -0.735240786278836, -0.000846576656319611;]

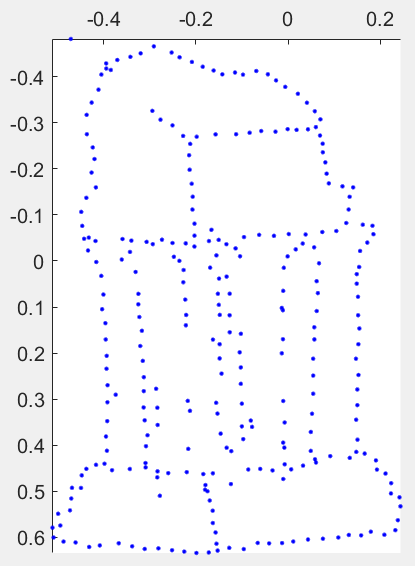
3.1.4:

camera2(E) will give fore candidate extrinsic matrices, I choose the extrinsic matrices with most positive Z, which has most points in front of images.

re-projection error using pts1: 0.5611

re-projection error using pts2: 0.5658

3.1.5:



3.2.1:

A picture containing indoor, building, dark, arch

Description automatically generated

3.2.2:

A picture containing diagram

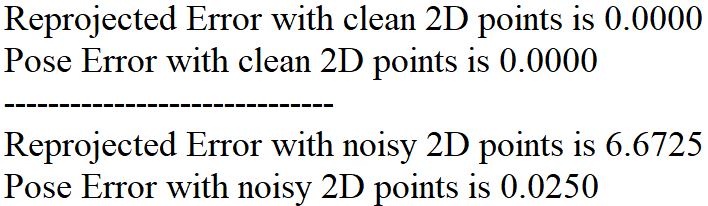
Description automatically generated

3.2.3:

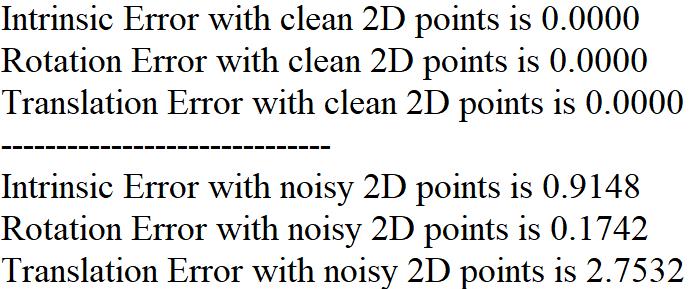
A picture containing graphical user interface

Description automatically generated

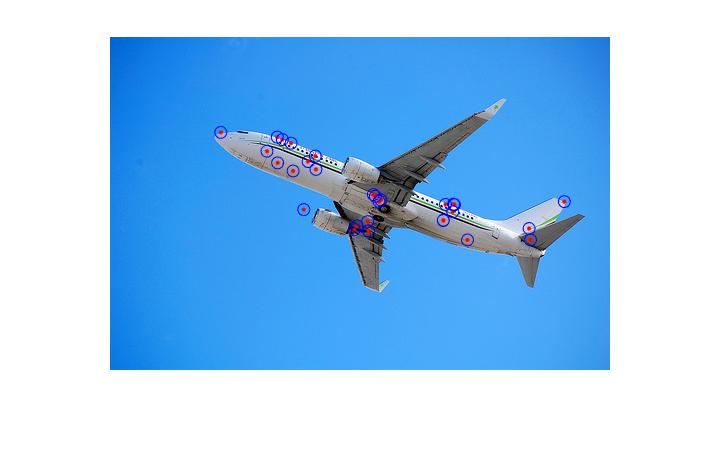
3.3.1:



3.3.2:



3.3.3:

  
A picture containing diagram

Description automatically generated

A picture containing aircraft, airplane, transport

Description automatically generated