# EE 8374: Fundamentals of Computer Vision

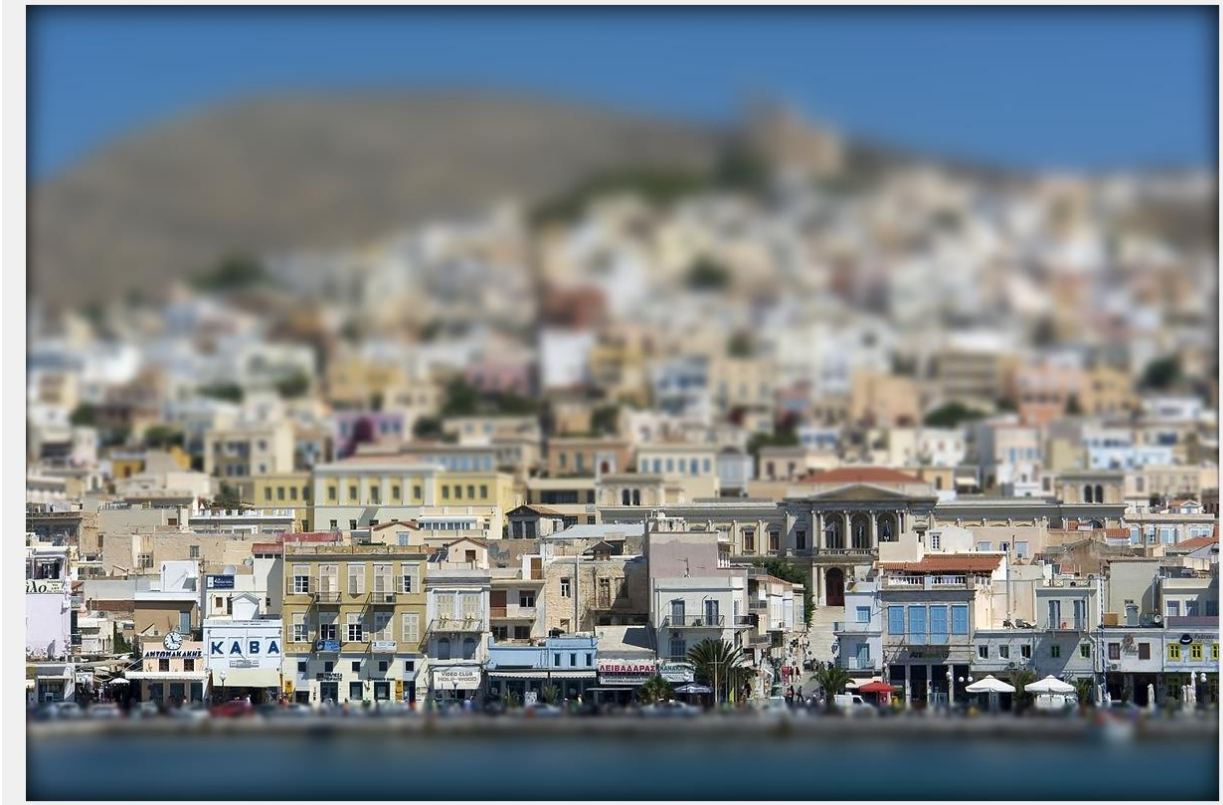
## Midterm Fall 2018


## Mingze Sun

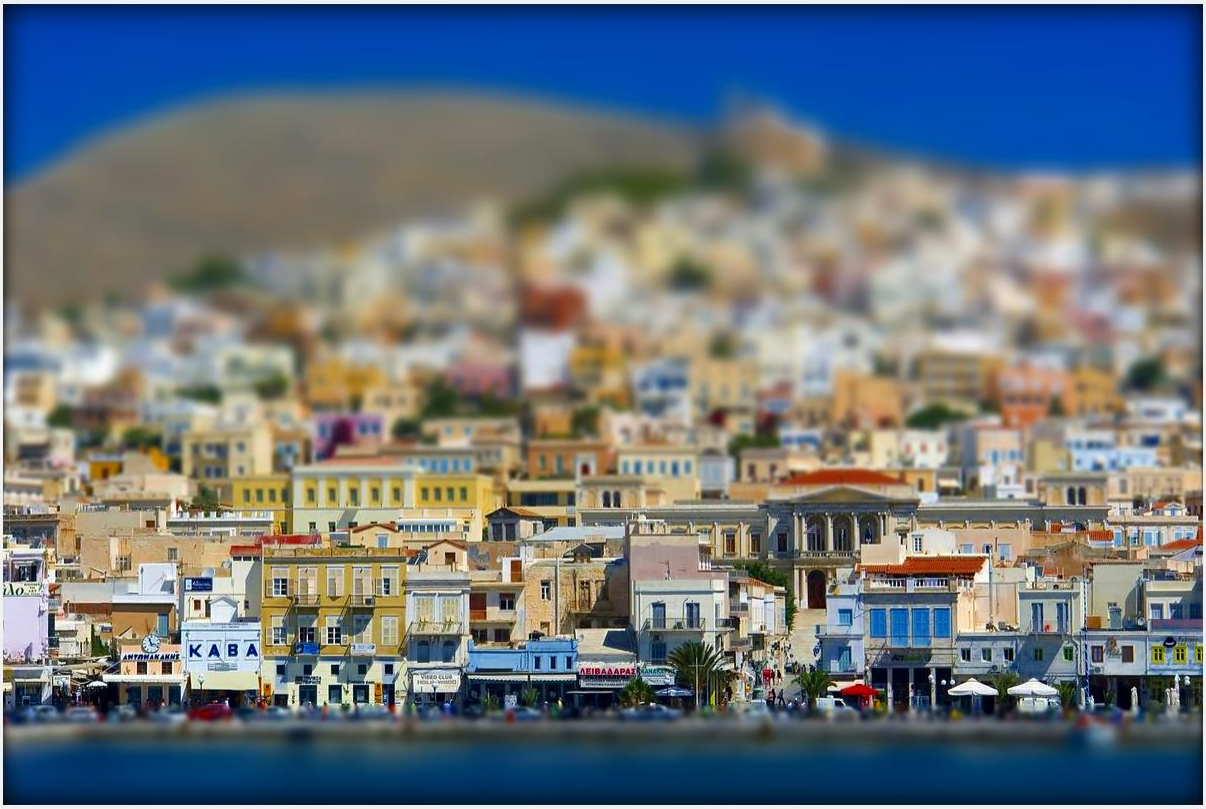

## Q1: Faking miniatures [25 points]

Original image:

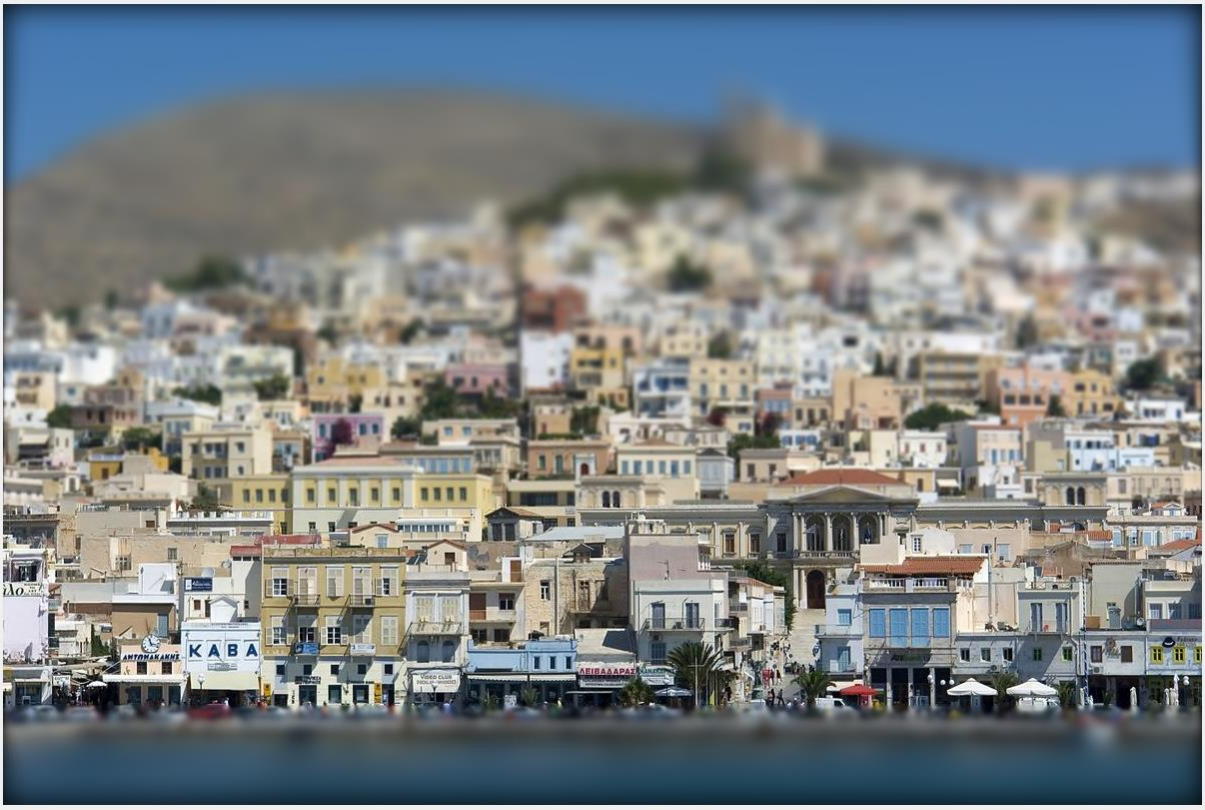

Fake miniature:




Fake miniature with saturation enhancing:




Nonlinear:

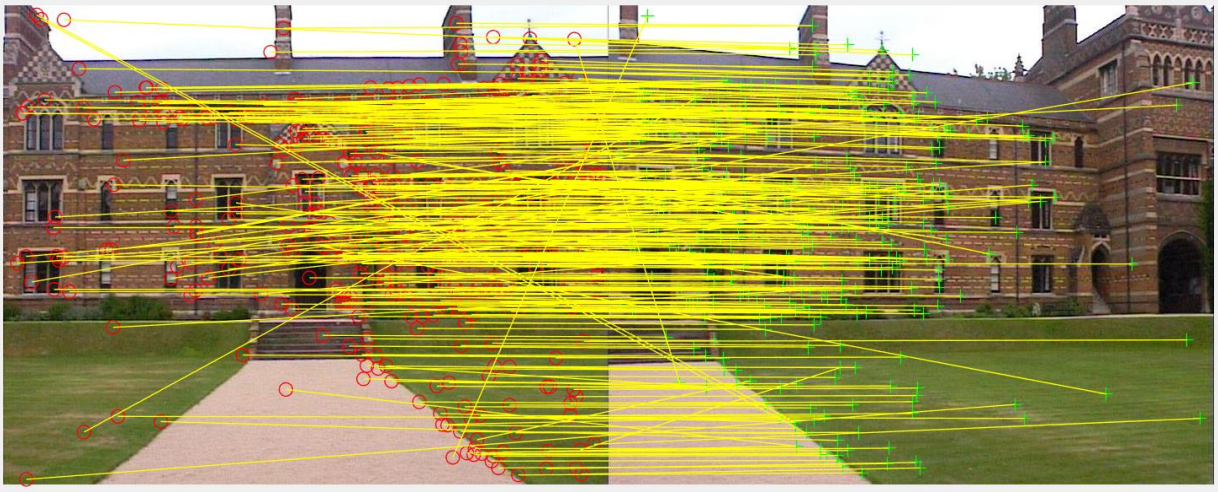


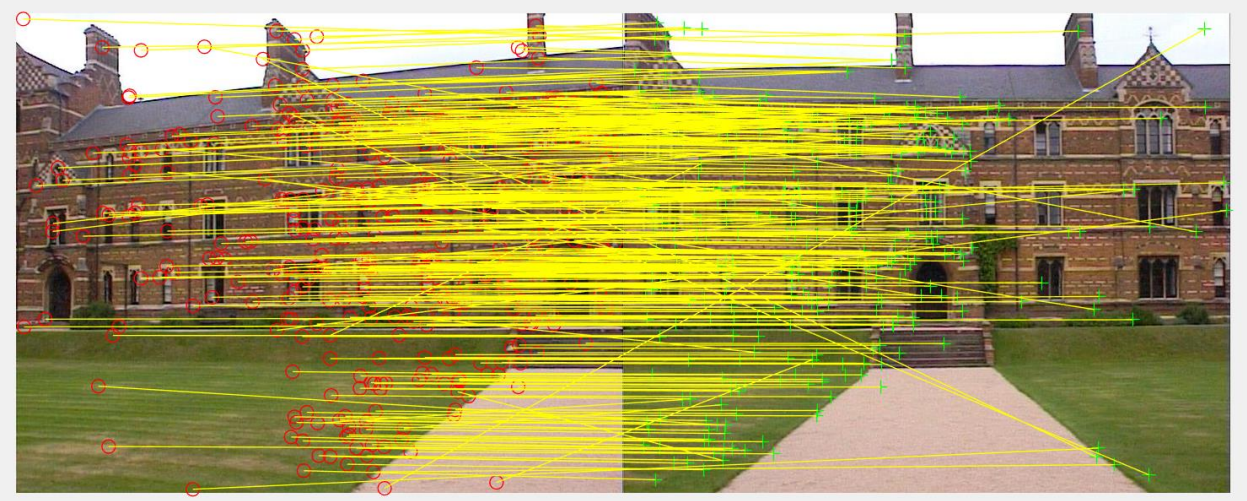

Nonlinear fake miniature with saturation enhancing with saturation enhancing:


# Q1: Faking miniatures [25 points]

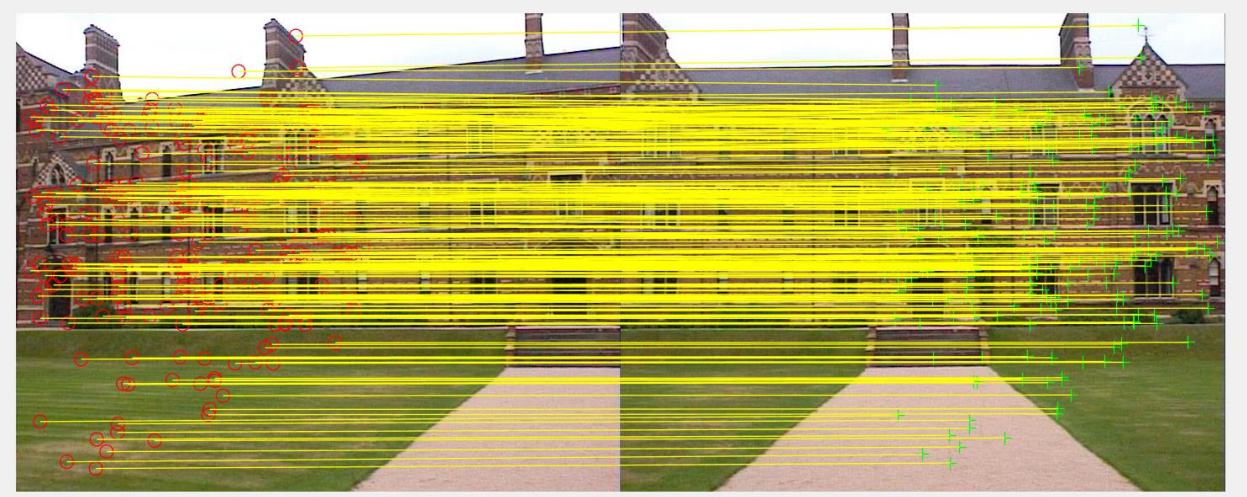
Problem1: match correspondences for each image pair

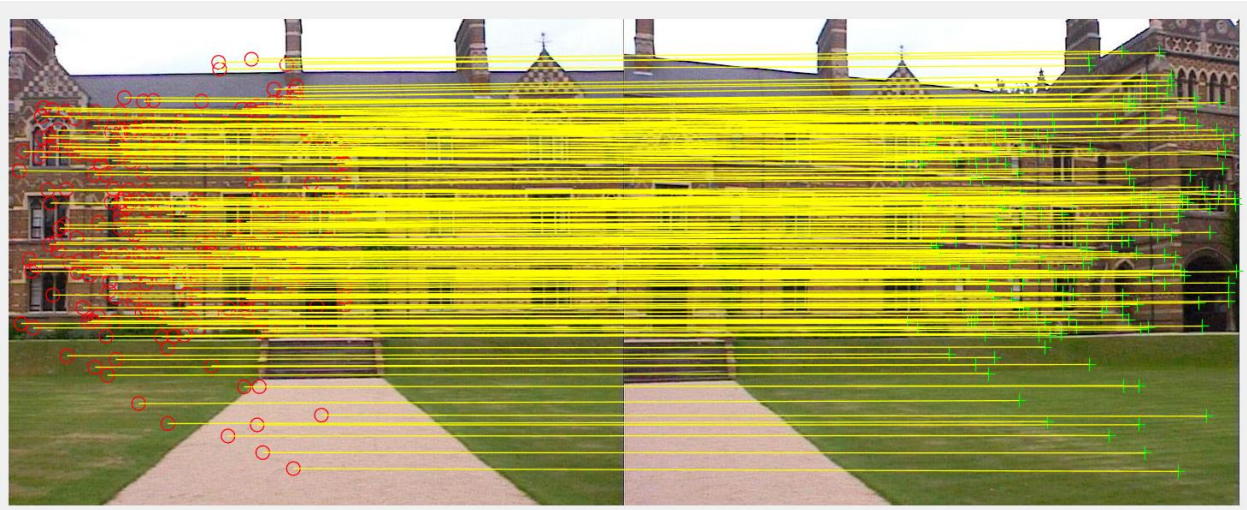
Correspondences image before using estimateGeometricTransform function:





Correspondences image after using estimateGeometricTransform function:





Problem 2:

image after homography:

Problem 3: Tform returns a 2-D geometric transform object, tform. The tform object maps the inliers inmatchedPoints1 to the inliers in matchedPoints2.

Problem 4: first 3x3 is a matrix with ones on the main diagonal and zeros elsewhere. The second one sets the property with a projective transformation from first to second image, the third one sets the property with a projective transformation from second to third image.

$$\underbrace{\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}}_{x'} \sim \underbrace{\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{x}$$

# Q3: Camera Calibration [50 points]

Problem 1: because we implement calibration with a plane, so we need to assume Zw = 0. Otherwise we will involve 3D, the distance between points on the plane cannot be estimate.

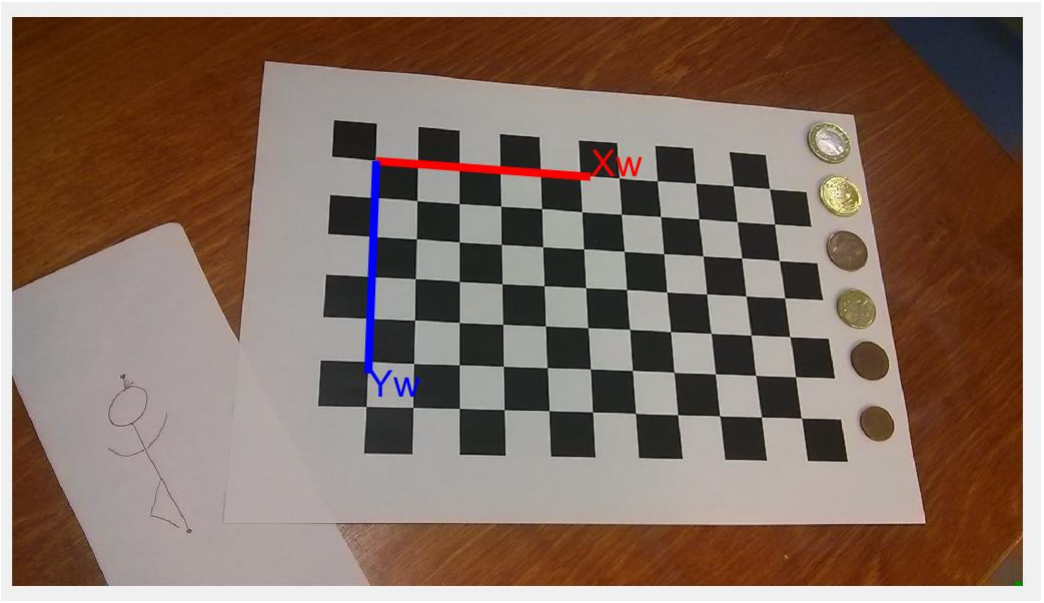
Problem 2: Estimated camera parameters accuracy, specified as an *M*-by-2-by-*P* array of [*x y*] coordinates. The [*x y*] coordinates represent the translation in *x* and *y* between the reprojected pattern key points and the detected pattern key points. The values of this property represent the accuracy of the estimated camera parameters.

Reprojection should be small if calibration is successful.

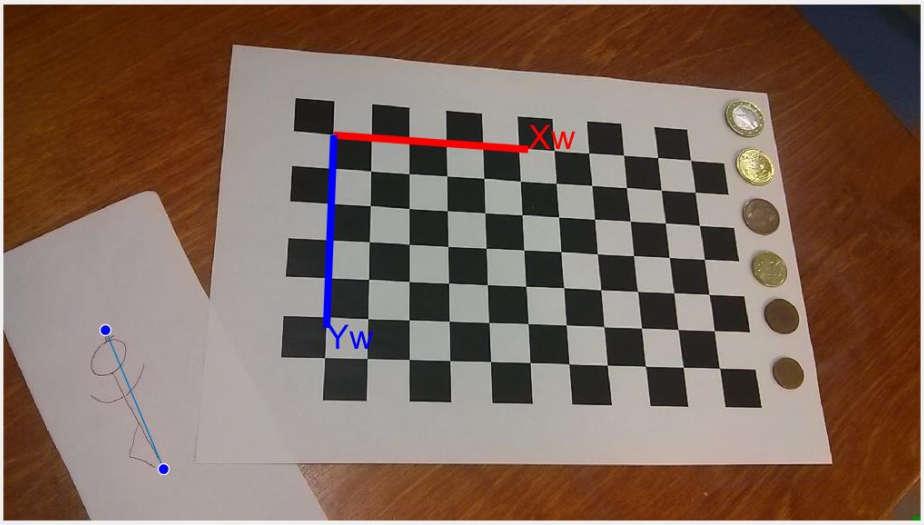
Problem 3:

Mathematic equation:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} [\boldsymbol{r}_1 \quad \boldsymbol{r}_2 \quad \boldsymbol{r}_3] \begin{bmatrix} 1 & 0 & 0 & -C_X \\ 0 & 1 & 0 & -C_Y \\ 0 & 0 & 1 & -C_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix}$$



Problem 4:



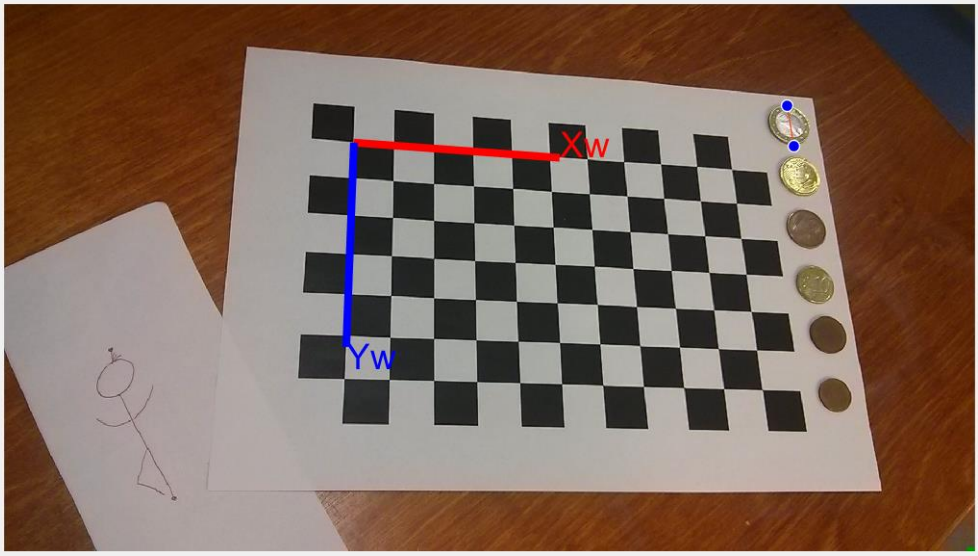

```
Command Window

D =

   66.0144

fx >>
```

Problem 5: it's larger than a single square. I use ginput to identify points in the figure, positioning the cursor with the mouse and calculate the Euclidean distance




Command Window

```
D =

   24.6051
```

Problem 6:

Image 1:

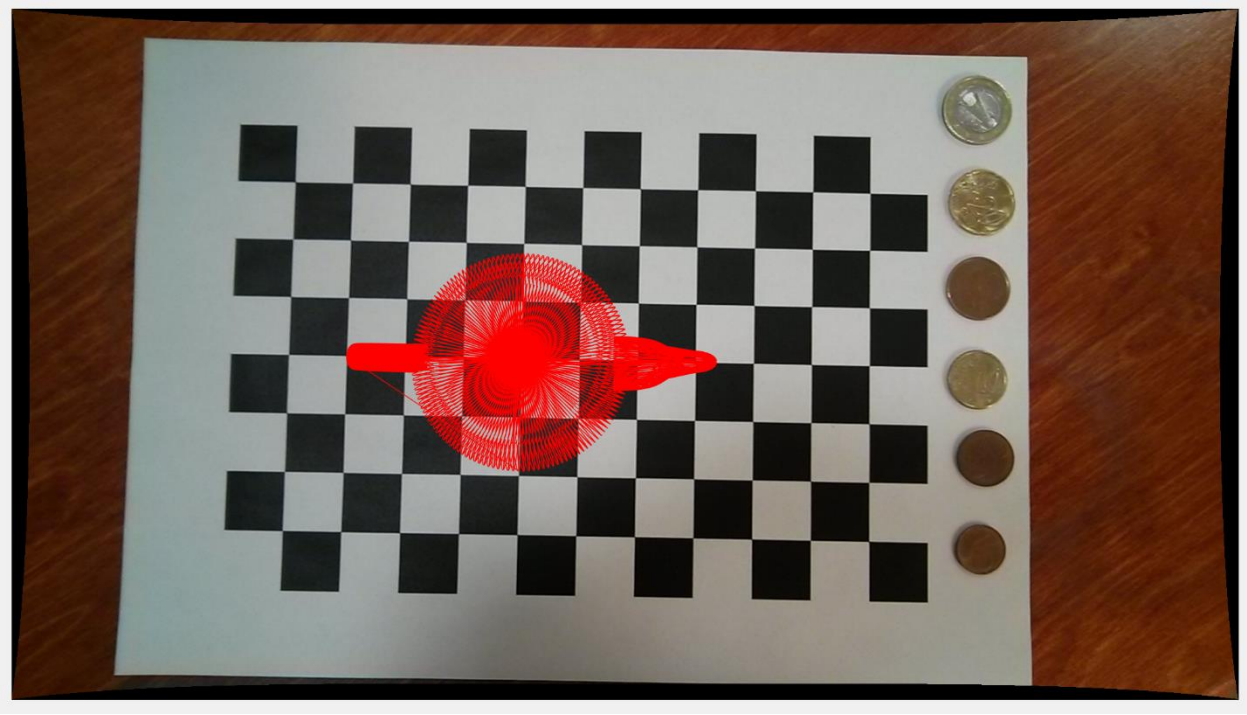


Image 2:

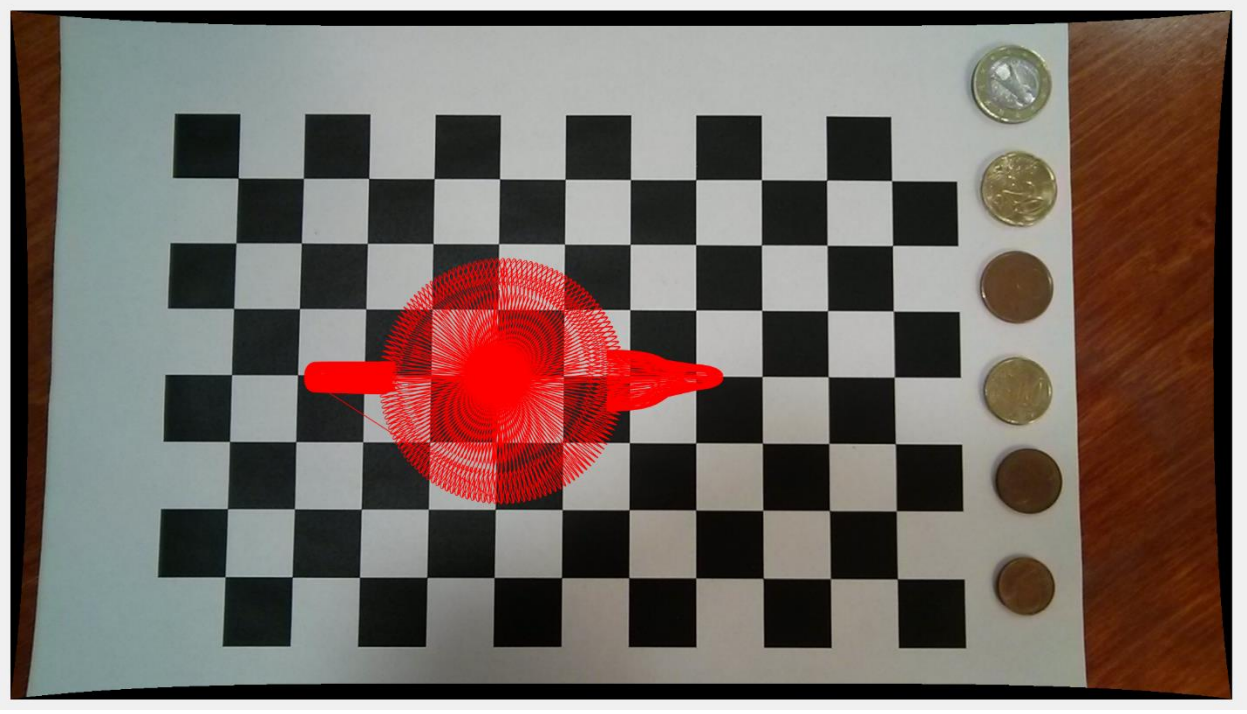


Image 3:

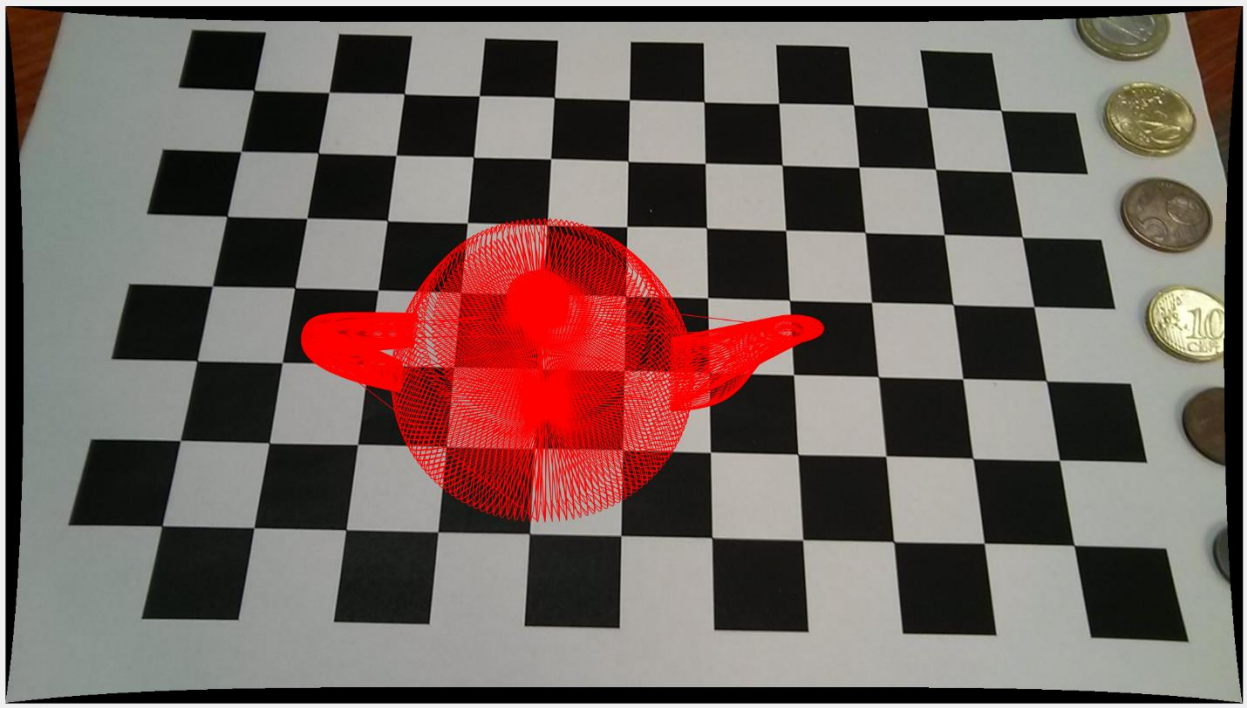


Image 4:

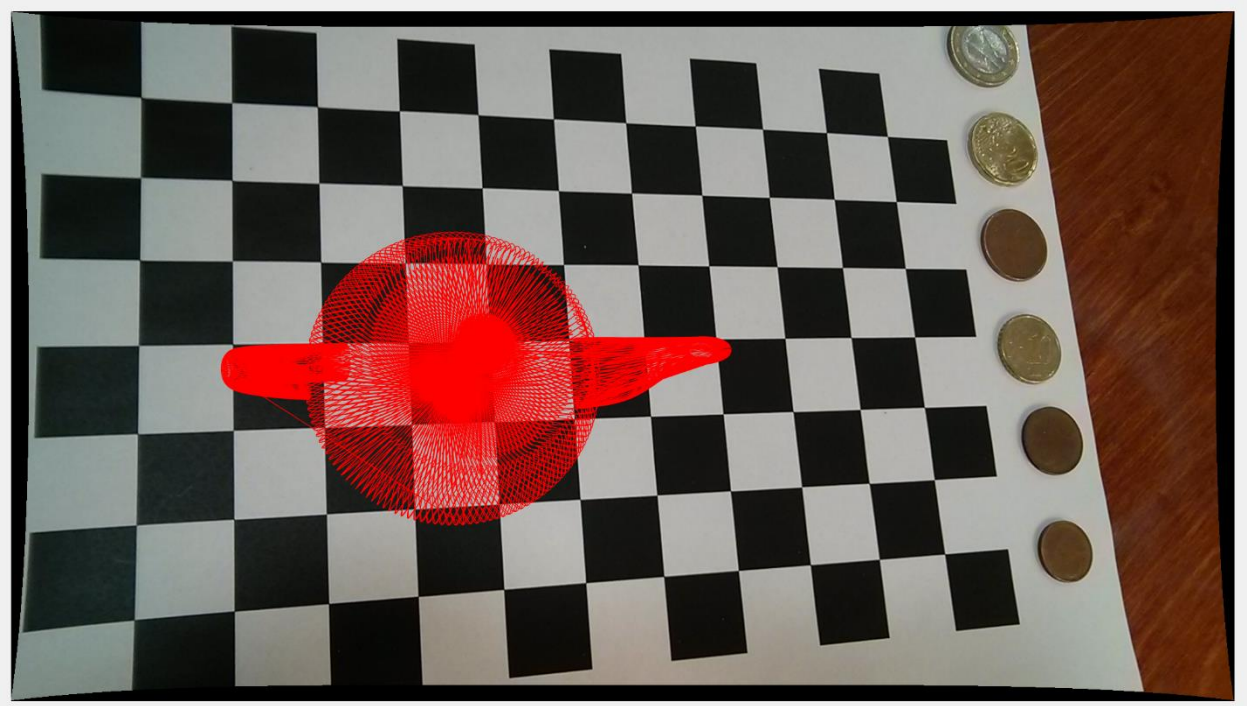


Image 5:

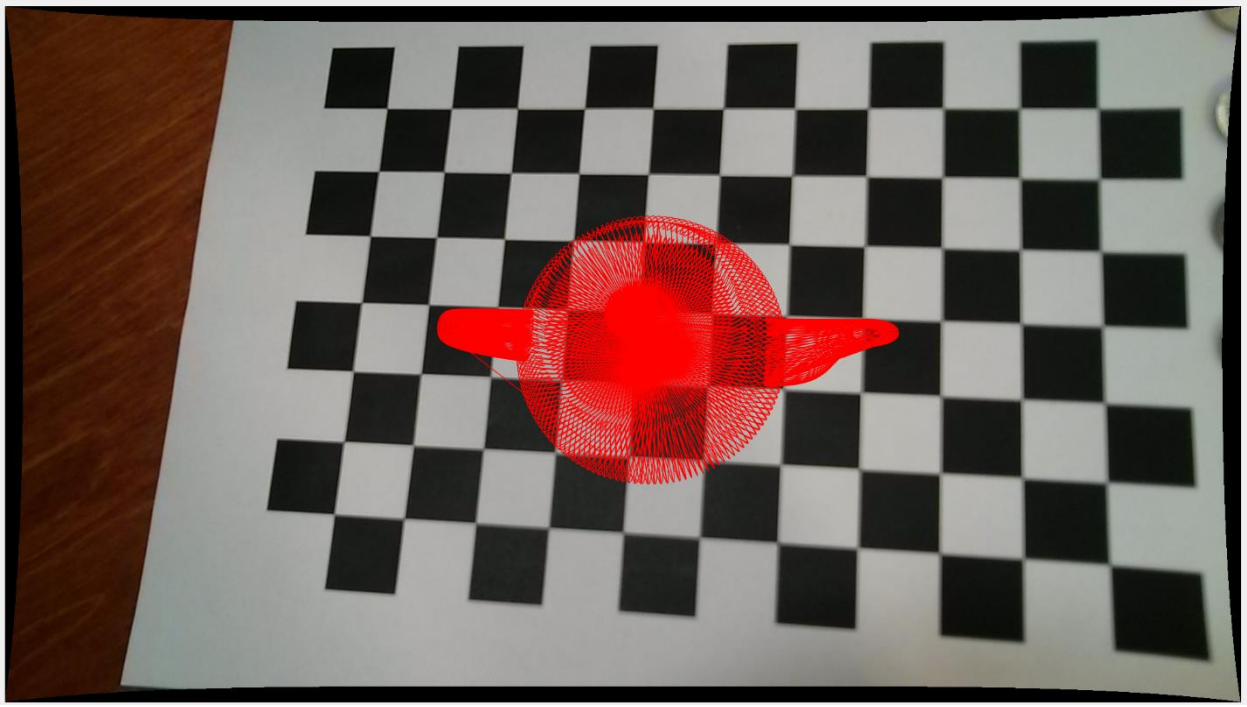


Image 6:

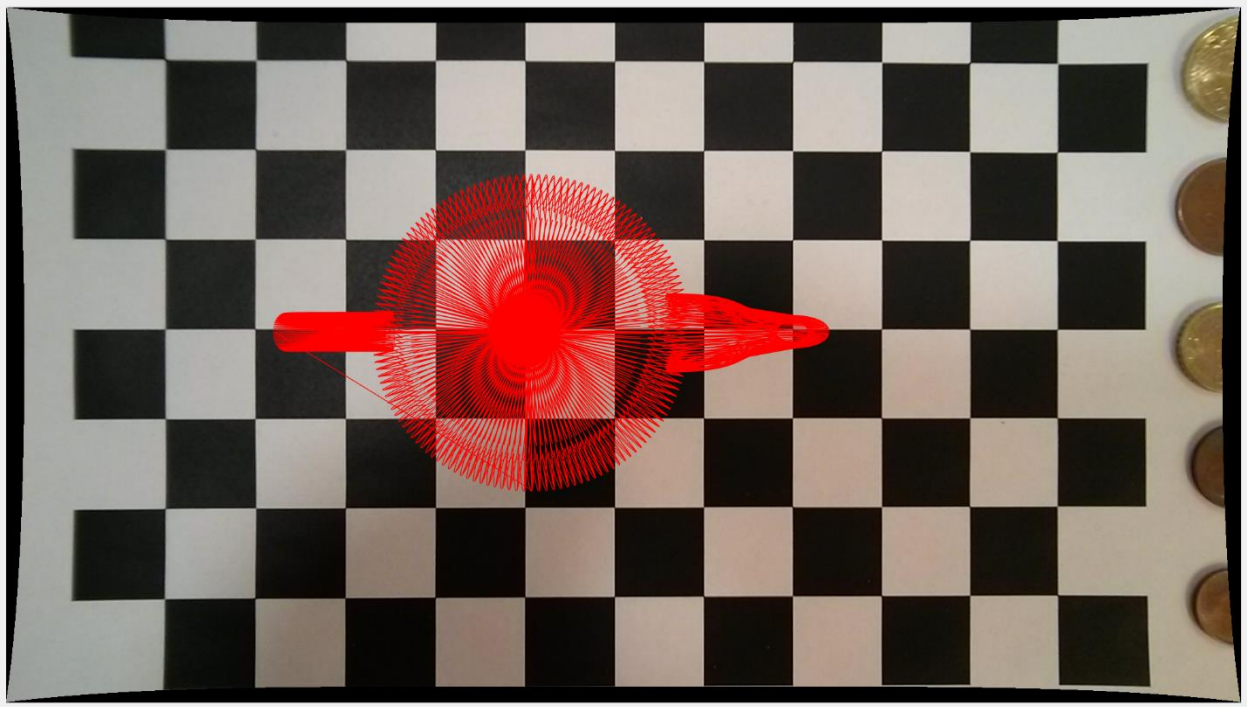


Image 7:

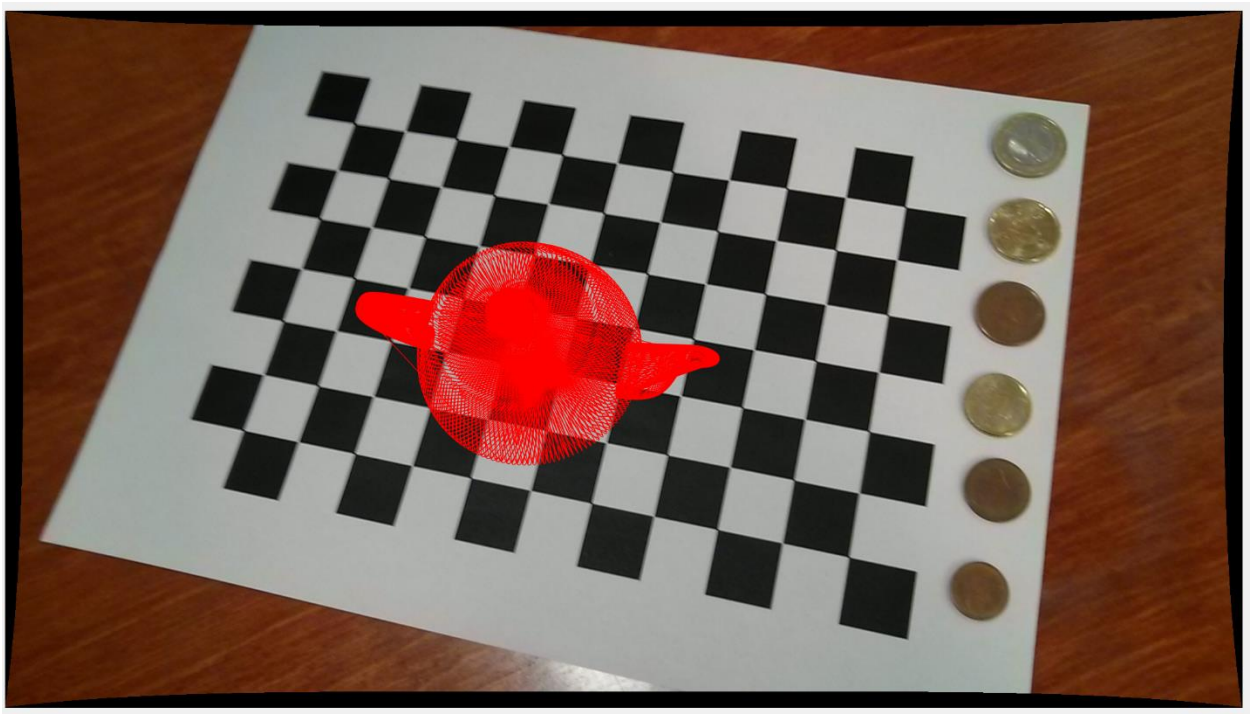


Image 8: