1. 实现张氏解法（编程***）

(*** Programming) Learn about Zhang's method [1] for camera calibration. Can you implement Zhang's method? Report the calibration result with your implemented approach. Compare with the result from Problem 6.

自学参考文献中的张氏解法并实现之。在报告中写下用自己方法实现的相机标定结果，并与任务6中结果做对比。

参考文献：

https://blog.csdn.net/weixin_43516894/article/details/88714407

https://blog.csdn.net/qq_xuanshuang/article/details/79639240

https://blog.csdn.net/weixin_41394379/article/details/85014100

OpenCV中calibrateCamera函数就是基于张正友方法实现的。这里自行实现张正友方法，由于张正友解法中涉及较多矩阵运算，这里采用Python语言编程。

图像坐标和物点坐标的关系可以表示为

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda K[R \quad | \quad t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (1)$$

K为内参矩阵，由于我们在后续的计算中将考虑径向和切向畸变，所以在内参矩阵中不体现径向畸变参数。

$$K = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (2)$$

外参矩阵可以分块为

$$[R \quad | \quad t] = [r_1 \quad r_2 \quad r_3 \quad t]$$

我们令图像平面位于世界坐标系的x0y平面，则物点Z轴坐标为0，所以可以改造（1）为

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda K[r_1 \quad r_2 \quad r_3 \quad t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda K[r_1 \quad r_2 \quad r_3 \quad t] \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda K[r_1 \quad r_2 \quad t] \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (3)$$

令

$$H = \lambda K[r_1 \quad r_2 \quad t] \cdots\cdots\cdots\cdots\cdots (4)$$

则有

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (5)$$

令

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (6)$$

联立（5,6）并改造，得

$$\begin{bmatrix} X & Y & 1 & 0 & 0 & 0 & -uX & -uY & -u \\ 0 & 0 & 0 & X & Y & 1 & -vX & -vY & -v \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0 \cdots\cdots\cdots\cdots\cdots (7)$$

物点坐标已知，像点坐标可以从图像中获取。故可扩展（7）为超定方程，使用SVD分解求解。

从一幅图像可得一个H矩阵。

然后使用最小二乘法优化H矩阵。程序中调用scipy.optimize库中的leastsq函数。误差函数的计算为，由实际物点坐标和H矩阵计算得到的理论像点坐标与实际像点坐标之差。并以此计算Jacobian矩阵。

误差计算为

$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \left(\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} - H \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \right) \cdots\cdots\cdots\cdots\cdots (8)$$

联立（6,8），得

$$\begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} u - \dfrac{h_{11}X + h_{12}Y + h_{13}}{h_{31} + h_{32} + h_{33}} \\ v - \dfrac{h_{21}X + h_{22}Y + h_{23}}{h_{31} + h_{32} + h_{33}} \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (9)$$

由（9）可以计算Δu和Δv对应hij的偏导数，然后构成Jacobian矩阵。

得到优化后的每幅图像的H矩阵后，可以按以下方式计算内参矩阵。

由于R是酉矩阵，且

$$R = [r_1 \quad r_2 \quad r_3]$$

则由酉矩阵的性质有

$$r_1{}^T r_2 = 0 \cdots\cdots\cdots\cdots\cdots (10)$$

$$\|r_1\| = \|r_2\| \cdots\cdots\cdots\cdots\cdots (11)$$

令

$$H = [h_1 \quad h_2 \quad h_3] \cdots\cdots\cdots\cdots\cdots (12)$$

由（4,12），得

$$r_i = K^{-1} h_i \cdots\cdots\cdots\cdots\cdots (13)$$

联立（10,11,13），得

$$h_1{}^T K^{-T} K^{-1} h_2 = 0 \cdots\cdots\cdots\cdots\cdots (14)$$

$$h_1{}^T K^{-T} K^{-1} h_1 = h_2{}^T K^{-T} K^{-1} h_2 \cdots\cdots\cdots\cdots\cdots (15)$$

令

$$B = K^{-T} K^{-1} \cdots\cdots\cdots\cdots\cdots (16)$$

联立（2,16），得

$$B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{\alpha^2} & 0 & -\dfrac{u_0}{\alpha^2} \\ 0 & \dfrac{1}{\beta^2} & -\dfrac{v_0}{\beta^2} \\ -\dfrac{u_0}{\alpha^2} & -\dfrac{v_0}{\beta^2} & \dfrac{u_0}{\alpha^2} + \dfrac{v_0}{\beta^2} + 1 \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (17)$$

B是实对称阵，有5个自由度，令向量b为

$$b = [B_{11} \quad B_{22} \quad B_{13} \quad B_{23} \quad B_{33}]^T \cdots\cdots\cdots\cdots\cdots\cdots (18)$$

联立（6,14,15,16,17,18），得

$$h_i{}^T B h_j = v_{ij} b \cdots\cdots\cdots\cdots\cdots (19)$$

其中v_{ij}为

$$v_{ij} = \begin{bmatrix} h_{i1}h_{j1} & h_{i2}h_{j2} & h_{i3}h_{j1} + h_{i1}h_{j3} & h_{i3}h_{j2} + h_{i2}h_{j3} & h_{i3}h_{j3} \end{bmatrix}^T \cdots\cdots\cdots\cdots\cdots (20)$$

联立（14,15,19），得

$$\begin{bmatrix} v_{12}{}^T \\ (v_{11} - v_{12})^T \end{bmatrix} b = 0 \cdots\cdots\cdots\cdots\cdots (21)$$

则由（21）和所有的图像的H矩阵，可以通过SVD分解求得b向量。

然后可以求得B矩阵，将B矩阵进行cholesky分解求得K，K即相机内参矩阵。

下一步根据内参矩阵、外参矩阵和每幅图像的H矩阵的关系可以求得外参矩阵。内参矩阵、外参矩阵和H矩阵的关系如（4），并且根据R矩阵为酉矩阵的性质可知

$$r_1 = \lambda K^{-1} h_1 = \frac{K^{-1} h_1}{\|K^{-1} h_1\|} \cdots\cdots\cdots\cdots\cdots (22)$$

$$r_2 = \lambda K^{-1} h_2 = \frac{K^{-1} h_2}{\|K^{-1} h_2\|} \cdots\cdots\cdots\cdots\cdots (23)$$

$$r_3 = r_1 \times r_2 \cdots\cdots\cdots\cdots\cdots (24)$$

$$\lambda = \frac{1}{\|K^{-1} h_1\|} = \frac{1}{\|K^{-1} h_2\|} \cdots\cdots\cdots\cdots\cdots (25)$$

到此，我们已经求出了相机内参矩阵和每幅图像的外参矩阵，接下来可以根据畸变规律求解径向畸变系数k1,k2,k3和切向畸变系数p1,p2.5个畸变系数与相机内参和外参的关系如下

在无畸变发生时，在相机坐标系中物点坐标为

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + t$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$r^2 = x'^2 + y'^2$$

$$x'' = x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x' y^1 + p_2(r^2 + 2x'^2) \cdots\cdots\cdots\cdots\cdots (26)$$

$$y'' = y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y'^2) + 2p_2 x' y^1 \cdots\cdots\cdots\cdots\cdots (27)$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda K \begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (28)$$

（28）表示的是考虑径向畸变和切向畸变后计算的像素点坐标。

令无畸变发生情况下理想像素点坐标为

$$\begin{bmatrix} \hat{u} \\ \hat{v} \\ 1 \end{bmatrix} = \lambda K [r_1 \quad r_2 \quad t] \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (29)$$

联立（2,26,27,28,29），得

$$\begin{bmatrix} (\hat{u} - u_0)r^2 & (\hat{u} - u_0)r^4 & 2\alpha x' y' & \alpha(r^2 + 2x'^2) & (\hat{u} - u_0)r^6 \\ (\hat{v} - v_0)r^2 & (\hat{v} - v_0)r^4 & \beta(r^2 + 2y'^2) & 2\beta x' y' & (\hat{v} - v_0)r^6 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ p_1 \\ p_2 \\ k_3 \end{bmatrix} = \begin{bmatrix} u - \hat{u} \\ v - \hat{v} \end{bmatrix} \cdots\cdots\cdots\cdots\cdots (30)$$

则根据所有图像的所有角点可以扩充（30）为超定方程，使用SVD分解求解得到畸变向量k

$$k = [k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3]^T$$

最后一步利用scipy.optimize库中的leastsq函数优化内参、外参和畸变向量。误差函数的计

算为所有图像的所有角点计算值与实际值的差距构成的列表。使用numpy的gradient函数求jacobian矩阵。

误差函数计算为

$$\begin{bmatrix}\Delta u\\ \Delta v\end{bmatrix}=\begin{bmatrix}u\\ v\end{bmatrix}-f(K,k,W)\cdots\cdots\cdots\cdots\cdots(31)$$

f()函数表示根据内参K、畸变向量k和外参W计算得到的理论像素坐标值，计算方式参考（26,27,28）

本工程根据附件1图片集实现的标定结果如下

```
total max error:
639.8024992752984
intrinsics_parm:
[[542.16211996   0.           353.73870404]
 [  0.           543.22833239 234.21803433]
 [  0.             0.           1.          ]]
distortionk:
[ 1.19186571e-01 -1.09194673e-01 -2.97420452e-05  1.76502572e-04
  5.71899425e-04]
extrinsics_parm:
[[[ 9.72931324e-01 -4.06266697e-03  2.31058723e-01 -1.01051405e+02]
  [ 2.34456618e-02  9.96421705e-01 -8.12039857e-02 -1.30622839e+02]
  [-2.29902022e-01  8.44232260e-02  9.69545140e-01  4.99070110e+02]]

 [[ 1.13452637e-01  9.79719062e-01  1.65163733e-01 -7.89998705e+01]
  [-7.50637142e-01  1.93429605e-01 -6.31766466e-01  9.99284255e+01]
  [-6.50901206e-01 -5.23024608e-02  7.57358616e-01  4.31011516e+02]]

 [[ 9.34898392e-01 -3.50674287e-01  5.47041199e-02 -5.60450495e+01]
  [ 3.16018356e-01  8.92646898e-01  3.21424820e-01 -1.20997183e+02]
  [-1.61546883e-01 -2.83212041e-01  9.45353661e-01  3.97314886e+02]]

 [[ 9.72591319e-01 -1.10897864e-02  2.32256633e-01 -1.26389378e+02]
  [-1.92514842e-02  9.91592015e-01  1.27963493e-01 -7.98102745e+01]
  [-2.31722911e-01 -1.28927467e-01  9.64200291e-01  4.10498004e+02]]

 [[ 2.11582116e-01 -9.72985854e-01  9.23663117e-02  6.21521228e+01]
  [ 8.58057732e-01  2.30171443e-01  4.59083909e-01 -1.36892628e+02]
  [-4.67942237e-01 -1.78783173e-02  8.83578196e-01  3.98112508e+02]]

 [[-1.22085263e-01 -9.58637136e-01  2.57118710e-01  1.93213998e+02]
  [ 9.86370244e-01 -1.45979603e-01 -7.59190114e-02 -7.69133413e+01]
  [ 1.10312871e-01  2.44345652e-01  9.63393104e-01  4.38859130e+02]]

 [[-2.90523543e-01 -8.76047976e-01  3.84884418e-01  1.37073797e+01]
  [ 9.56860231e-01 -2.64378367e-01  1.20509661e-01 -8.47299578e+01]
  [-3.81713056e-03  4.03291487e-01  9.15063608e-01  4.71507757e+02]]

 [[-2.34762827e-01 -9.50961154e-01  2.01393392e-01  8.69340092e+01]
  [ 9.17921865e-01 -1.48705812e-01  3.67839682e-01 -1.04398931e+02]
  [-3.19852881e-01  2.71218482e-01  9.07818633e-01  3.94354770e+02]]

 [[ 8.96465932e-01 -1.67332281e-01 -4.10303229e-01 -8.72112907e+01]
  [ 9.19285787e-02  9.76043249e-01 -1.97202212e-01 -9.69350784e+01]
  [ 4.33471993e-01  1.39066472e-01  8.90372140e-01  3.48823814e+02]]
```

```
 [[ 1.60245483e-01 -7.96373986e-01 -5.83189386e-01  4.79864783e+01]
  [ 9.82027298e-01  1.88312082e-01  1.26864364e-02 -1.32182782e+02]
  [ 9.97184596e-02 -5.74740842e-01  8.12237154e-01  4.22494024e+02]]

 [[ 8.80678716e-03 -9.95344011e-01  9.59830240e-02  5.26648569e+01]
  [ 9.28575331e-01  4.37565748e-02  3.68555583e-01 -1.21774608e+02]
  [-3.71039480e-01  8.58816776e-02  9.24637249e-01  3.99224888e+02]]

 [[ 3.05031899e-01 -9.51766966e-01  3.30935425e-02  3.31258483e+01]
  [ 8.38052585e-01  2.51757498e-01 -4.84031018e-01 -1.09263213e+02]
  [ 4.52353186e-01  1.75379029e-01  8.74424834e-01  3.63203848e+02]]

 [[ 1.44906346e-01 -8.93856382e-01 -4.24291079e-01  4.62254985e+01]
  [ 9.63571651e-01  2.24907231e-01 -1.44728749e-01 -1.28911333e+02]
  [ 2.24792847e-01 -3.87862741e-01  8.93885155e-01  3.90662542e+02]]]
```

使用OpenCV库函数cameraCalibrate标定的结果为

```
Reprojection error:
0.4079421853068828
intrinsics_parm:
[[536.06448455   0.         342.36860717]
 [  0.         536.00716267 235.53171828]
 [  0.           0.           1.        ]]
distortionk:
[[-0.26511883 -0.04659248  0.00183174 -0.00031504  0.25213778]]
extrinsics_parm:
[[ 9.62219428e-01  9.80020974e-03  2.72098748e-01 -9.03338831e+01]
 [ 3.62681653e-02  9.85833041e-01 -1.63761518e-01 -1.30722374e+02]
 [-2.69848833e-01  1.67443036e-01  9.48232270e-01  4.79779403e+02]]

[[ 9.76190611e-02  9.75901770e-01  1.95157001e-01 -7.03632993e+01]
 [-7.56827579e-01  2.00134354e-01 -6.22220424e-01  9.95810464e+01]
 [-6.46283634e-01 -8.69596271e-02  7.58126301e-01  4.24613834e+02]]

[[ 9.21161708e-01 -3.66323695e-01  1.31407983e-01 -4.78732644e+01]
 [ 3.15574759e-01  9.00678867e-01  2.98647199e-01 -1.20476988e+02]
 [-2.27757939e-01 -2.33633322e-01  9.45273396e-01  3.81885300e+02]]

[[ 9.71424468e-01 -1.10982938e-02  2.37089290e-01 -1.18150414e+02]
 [-1.53196807e-02  9.93891419e-01  1.09293893e-01 -8.07687106e+01]
 [-2.36853987e-01 -1.09802894e-01  9.65320420e-01  3.97126884e+02]]

[[ 1.94784023e-01 -9.71117261e-01  1.37805840e-01  7.01312823e+01]
 [ 8.65521758e-01  2.36274713e-01  4.41640517e-01 -1.38358823e+02]
 [-4.61444764e-01  3.32494361e-02  8.86545658e-01  3.80717706e+02]]

[[-8.98315251e-02 -8.96139994e-01  4.34584179e-01  2.00644983e+02]
 [ 9.92181399e-01 -1.18480386e-01 -3.92233421e-02 -7.86571840e+01]
 [ 8.66393067e-02  4.27662846e-01  8.99776706e-01  4.03879777e+02]]

[[-3.19688022e-01 -9.00930211e-01  2.93469459e-01  2.33654008e+01]
 [ 9.46288886e-01 -2.87770370e-01  1.47395926e-01 -8.61555326e+01]
 [-4.83416281e-02  3.24827599e-01  9.44536986e-01  4.67400370e+02]]

[[-2.43600385e-01 -9.49994258e-01  1.95370834e-01  9.47994354e+01]
```

```
 [ 9.17153651e-01 -1.60125450e-01  3.64950711e-01 -1.05508816e+02]
 [-3.15417237e-01  2.68087207e-01  9.10297323e-01  3.80094129e+02]]

[[ 9.03225024e-01 -1.69429921e-01 -3.94307060e-01 -7.96631488e+01]
 [ 8.50716487e-02  9.71224255e-01 -2.22455076e-01 -9.72012887e+01]
 [ 4.20651126e-01  1.67382640e-01  8.91647734e-01  3.34050767e+02]]

[[ 1.57146409e-01 -8.08440302e-01 -5.67211851e-01  5.62154989e+01]
 [ 9.82184196e-01  1.87870499e-01  4.34517804e-03 -1.33180775e+02]
 [ 1.03049556e-01 -5.57789345e-01  8.23560462e-01  4.05771226e+02]]

[[ 5.98197500e-03 -9.97406019e-01  7.17317849e-02  6.08575364e+01]
 [ 9.30486309e-01  3.18266084e-02  3.64941497e-01 -1.23095984e+02]
 [-3.66277825e-01  6.45623728e-02  9.28263031e-01  3.86737271e+02]]

[[ 3.08605419e-01 -9.50293990e-01  4.12798663e-02  4.03781946e+01]
 [ 8.38022532e-01  2.51102690e-01 -4.84423032e-01 -1.09974186e+02]
 [ 4.49978811e-01  1.84089031e-01  8.73859427e-01  3.49991932e+02]]

[[ 1.46285674e-01 -8.94998790e-01 -4.21399652e-01  5.39579478e+01]
 [ 9.62354938e-01  2.27393301e-01 -1.48880019e-01 -1.29789684e+02]
 [ 2.29070894e-01 -3.83757022e-01  8.94570888e-01  3.75036833e+02]]
```

标定的内参矩阵和外参矩阵基本一致，畸变向量有较大出入。猜测这是因为本工程是完全基于张正友标定方法的，而系统库函数是基于张正友标定方法和Bouguet极线校正的方法BouguetMCT工具，由于使用工具和方法有所差异造成的。