Lista 6 – Visão Computacional

Aluno: Rennan de Lucena Gaio DRE: 119122454

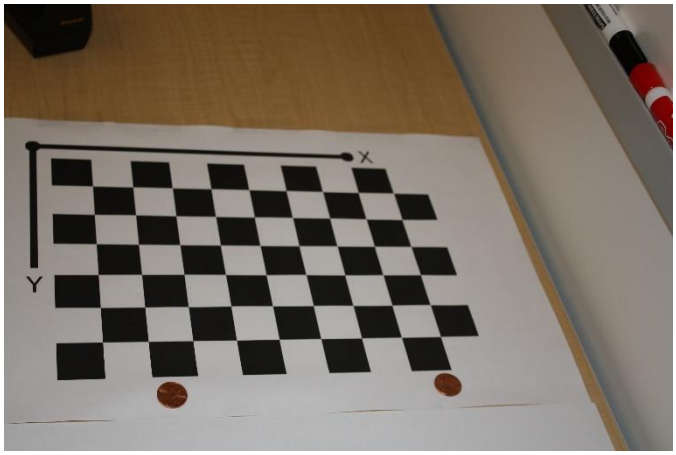
Todo o código do trabalho pode ser acessado pelo link do github:

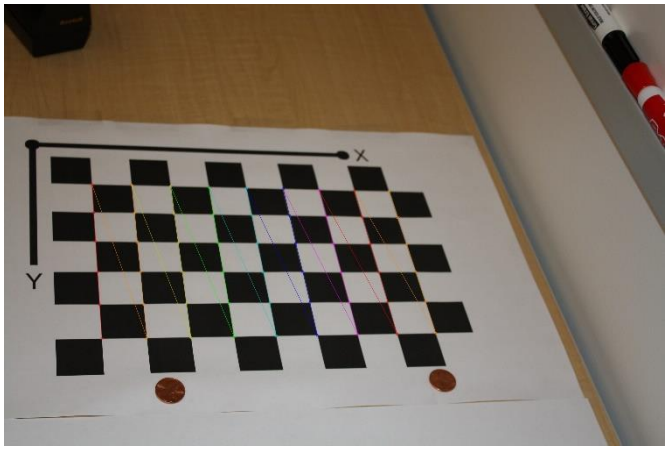
https://github.com/RennanGaio/visao_computacional/tree/master/lista6

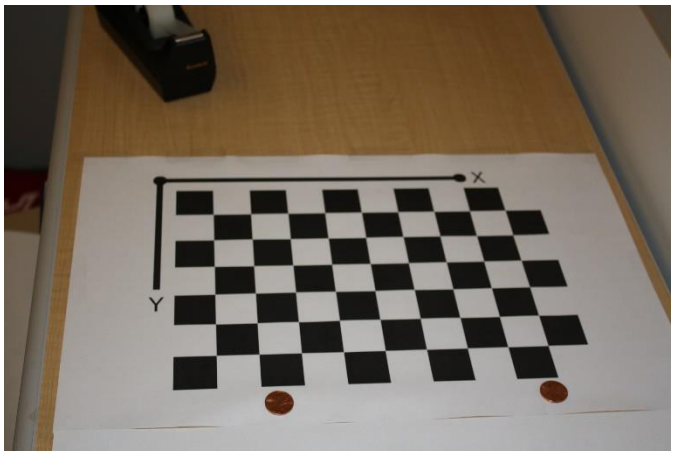
Calibração de Zhang e distorção radial

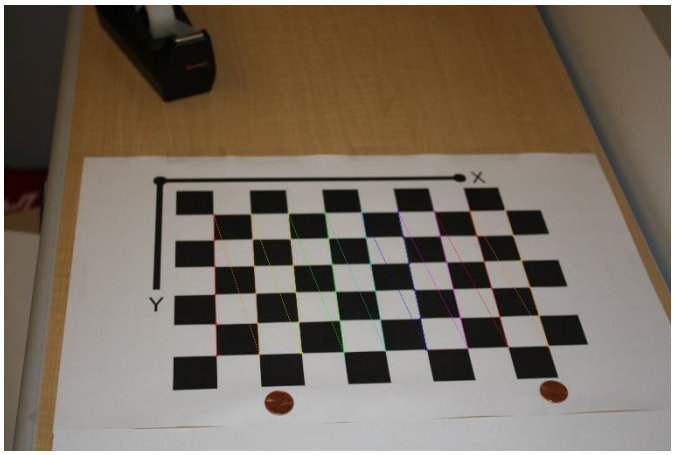
1. Dataset 1 “image_calib_X”

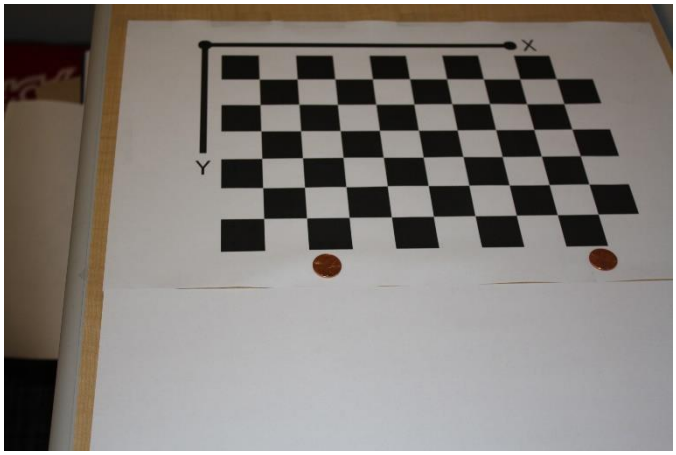
Para cada uma das imagens, foram detectadas todas as quinas do tabuleiro de xadrez utilizando o método em Python “findChessboardCorners” semelhante à função em matlab “detectCheckerboardPoints”. Tal abordagem foi escolhida pelo fato dos resultados serem muito superiores aos obtidos utilizando o método de Harris e SIFT, pois estes apresentavam alguns outliers que atrapalhavam as contas. Nas imagens abaixo, teremos um comparativo lado a lado das imagens originais e das imagens com os pontos dos cantos marcados em cada uma das imagens. Estes resultados também podem ser encontrados na pasta “/results” do repositório.

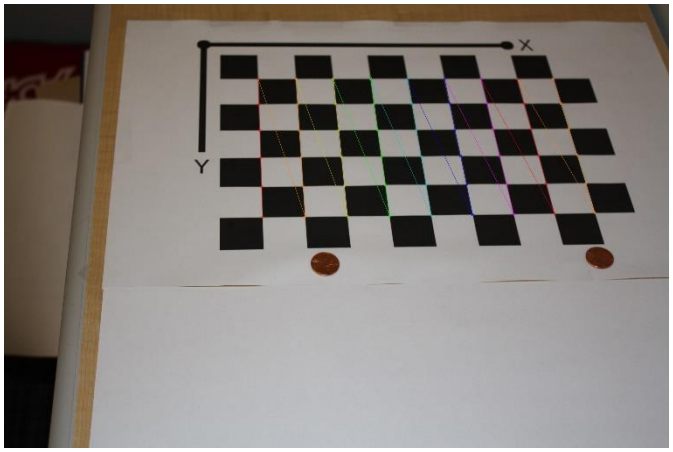


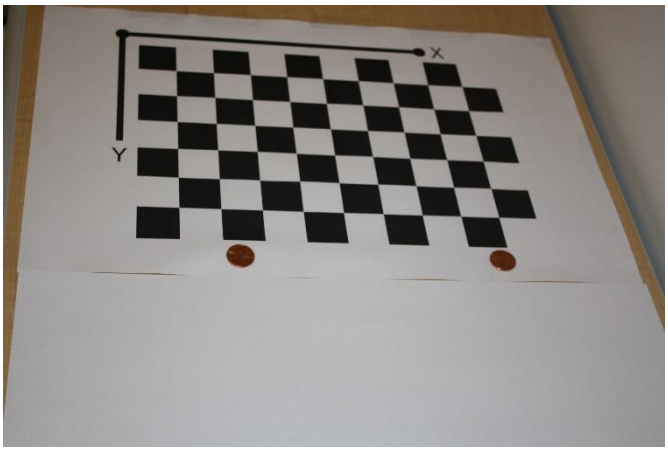


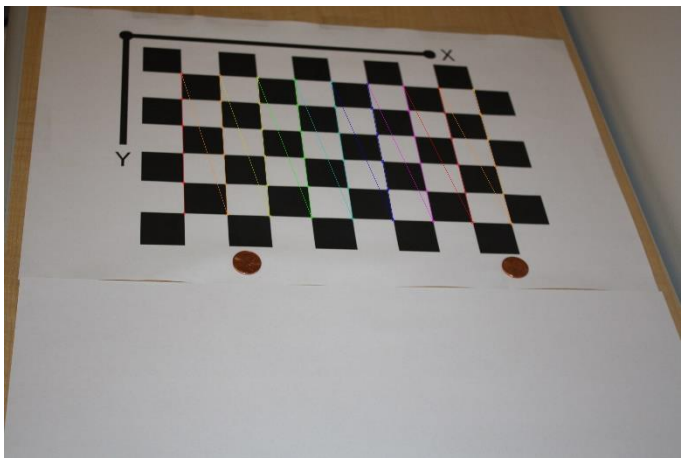

X
Y

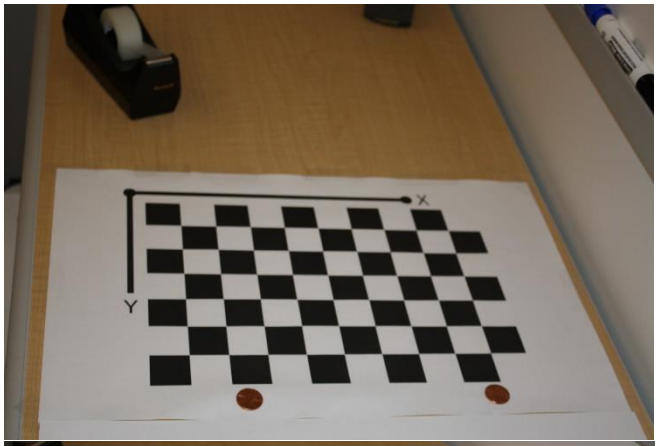
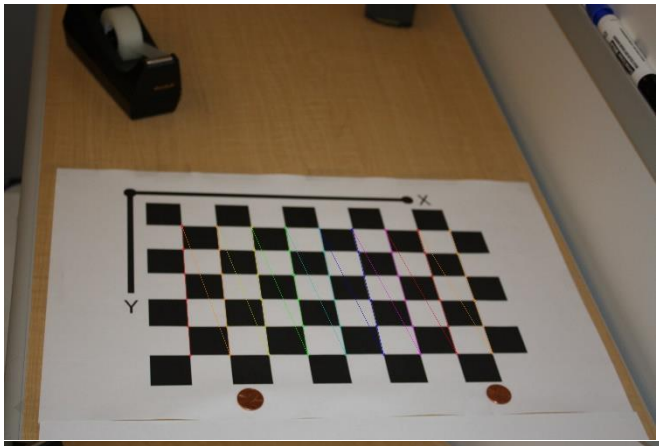
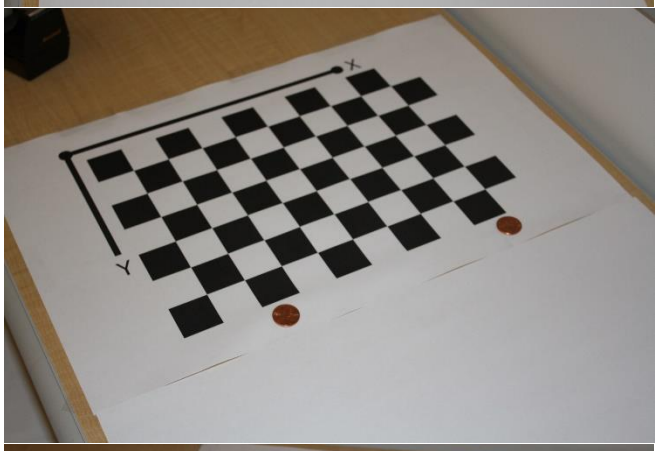
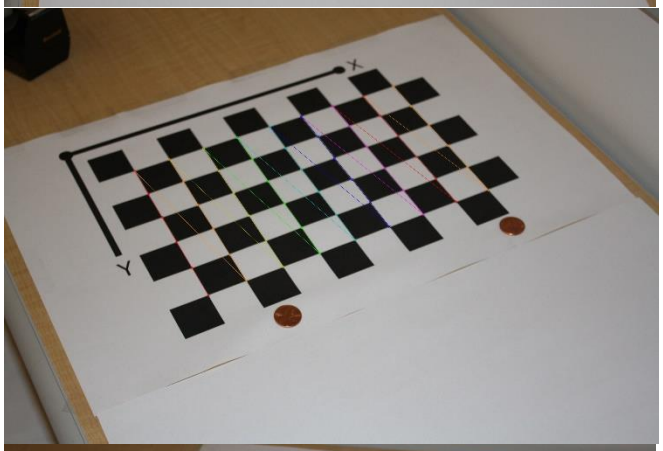
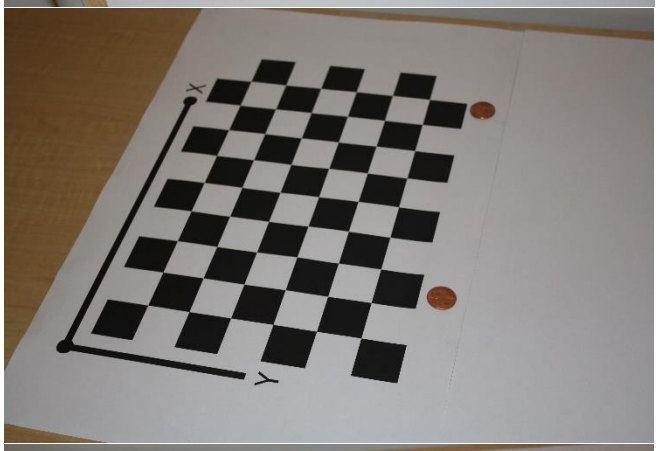
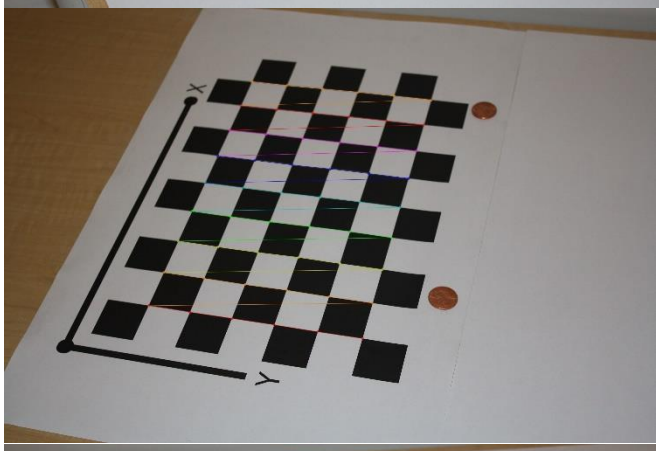
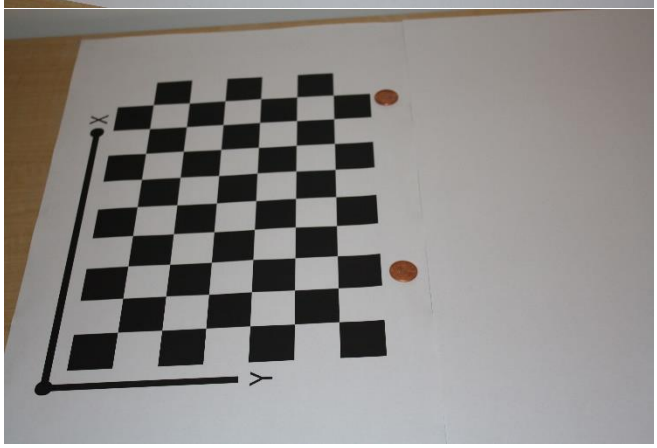
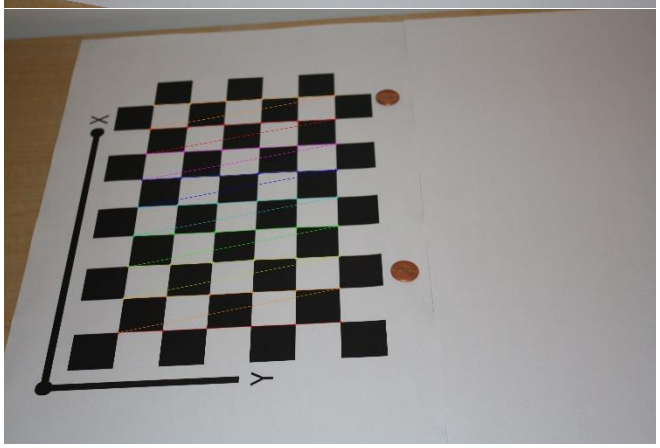

X
Y

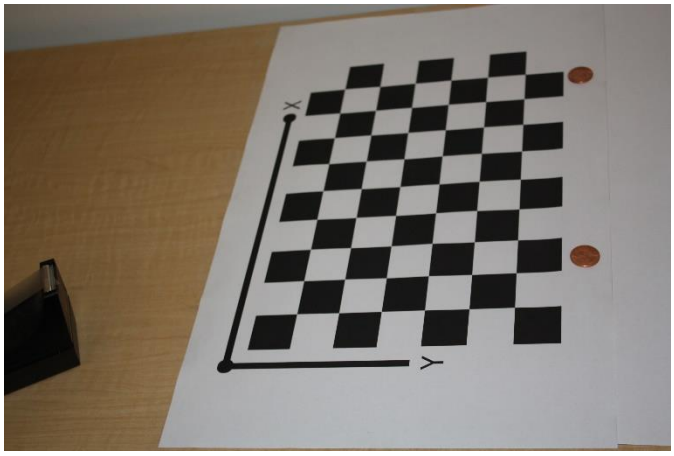

X
Y

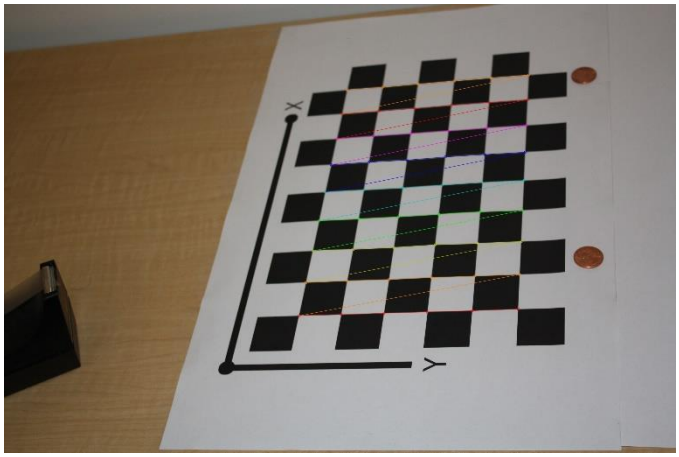

X
Y


X
Y


X
Y





Os pontos detectados estão coloridos ligados por uma linha sequencial entre eles para facilitar a visualização.

Para a representação do tabuleiro no mundo real, foi construída uma matriz representando os cantos do tabuleiro de xadres (contendo os 54 cantos interiores do tabuleiro). Esta matriz levou em consideração o tamanho dos quadrados de 29x29. Esta matriz foi chamada de "check_board_points" no código e está devidamente destacada em uma seção do notebook python.

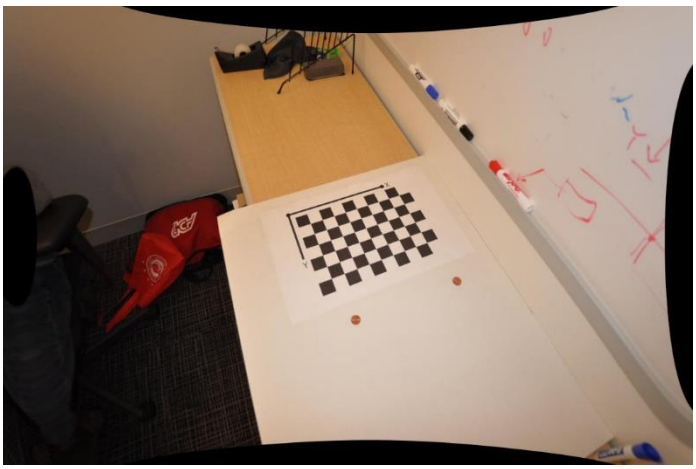
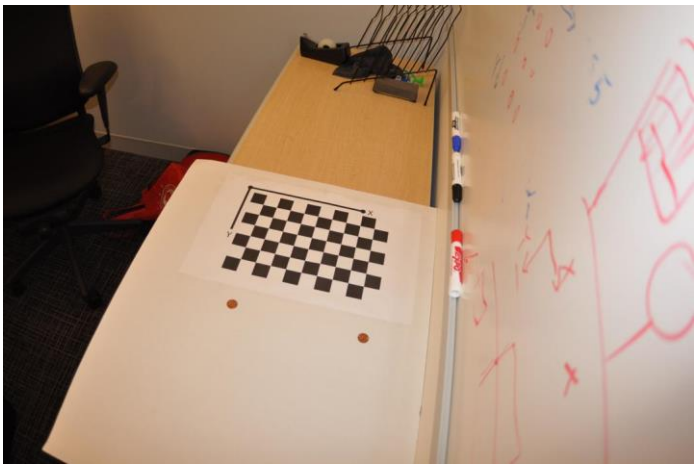
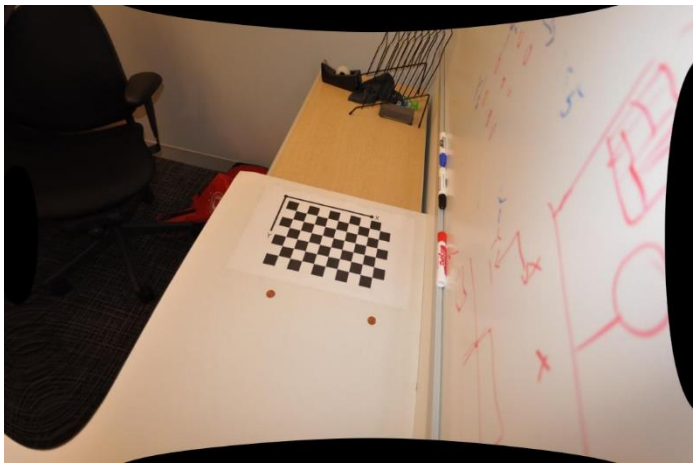
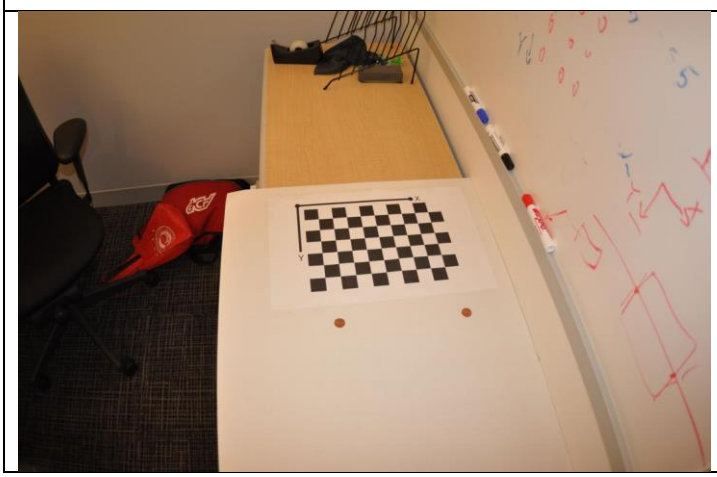
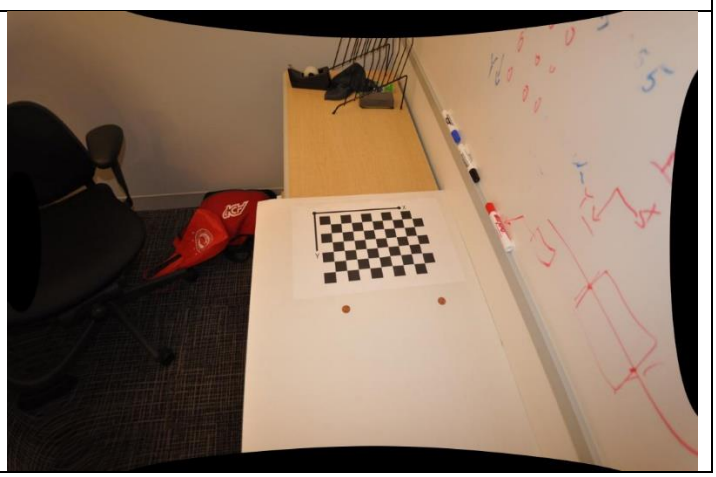
A partir das bordas de todas as imagens e da representação no mundo real dos pontos, foi possível fazer a calibração da câmera utilizando a função "cv2.calibrateCamera" do python. Esta função aplica o método de Zhang de calibração e como resultado é possível obter a matriz de parâmetros intrínsecos, os coeficientes de distorção, os vetores de rotação e de translação da imagem (parâmetros extrínsecos). Como foi utilizado python para a resolução da lista, o opencv não possui o método "showExtrinsics" do matlab, logo foi exibido os valores dos parâmetros extrínsecos obtidos na forma de matriz. O print dos resultados pode ser visto a seguir no documento, porém também podem ser observados na **janela 6 do python notebook**.

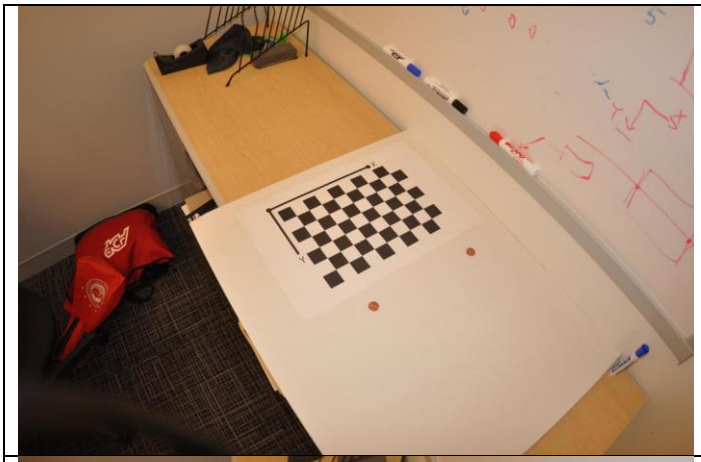
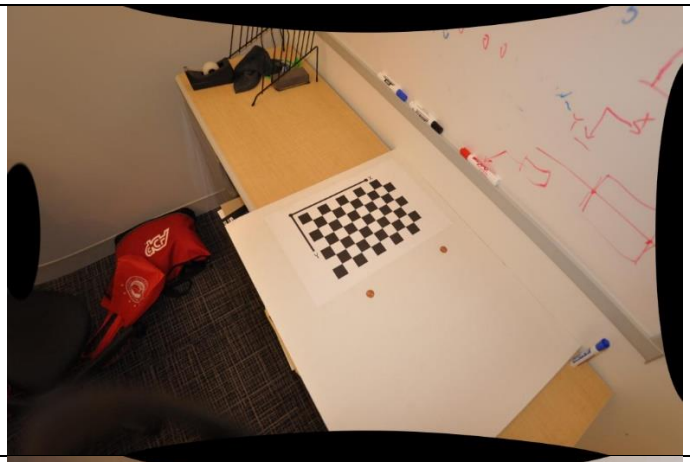
```
Matriz intrinseca da câmera :

[[4.63510957e+03 0.00000000e+00 1.46278207e+03]
 [0.00000000e+00 4.61457265e+03 9.51147015e+02]
 [0.00000000e+00 0.00000000e+00 1.00000000e+00]]


coeficiente de distorção da lente :

[[-1.27382329e-01  3.56180003e+00  1.34915824e-03  3.22295935e-03
  -3.46538251e+01]]
```

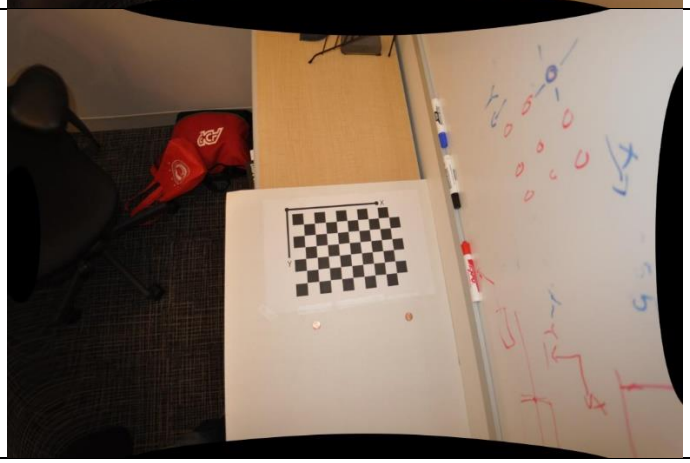
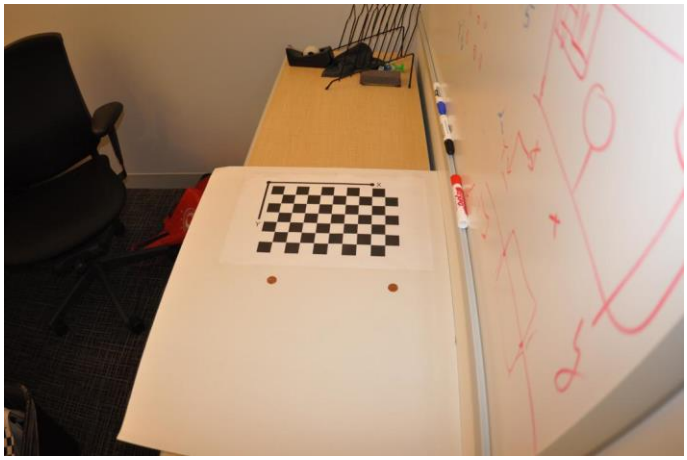
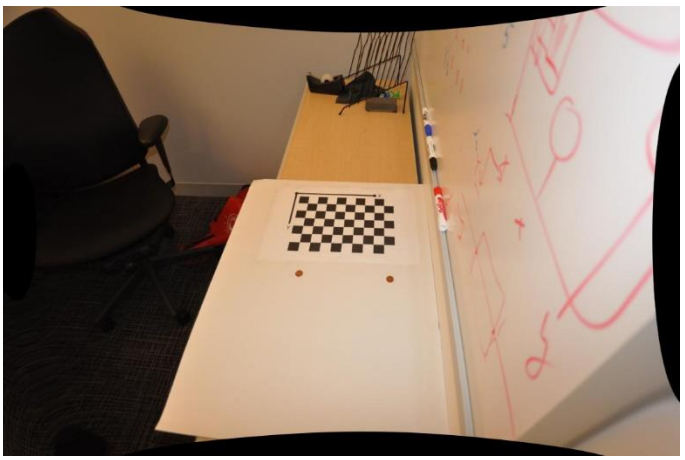
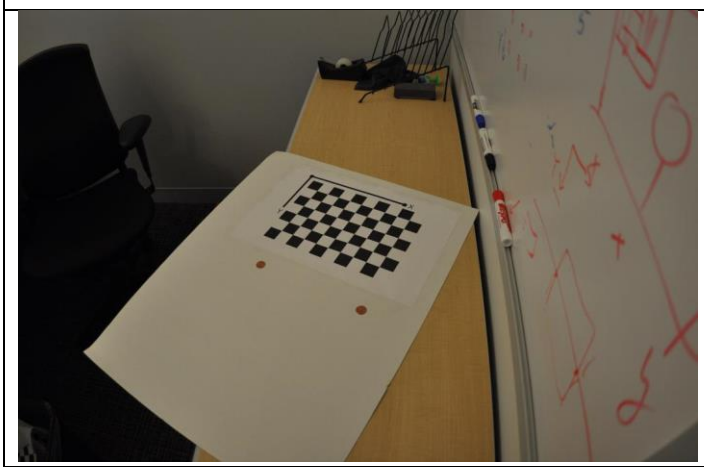
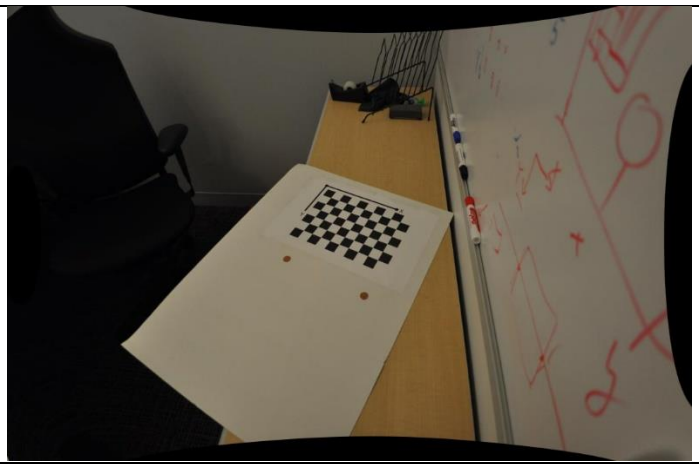
```
vetores de translação por imagem:

[array([[-158.7252822 ],
       [  71.77645982],
       [ 699.9307315 ]]), array([[-90.84188085],
       [ 84.91958786],
       [742.59217304]]), array([[-59.07151193],
       [ -7.763983  ],
       [742.69415184]]), array([[-115.53139161],
       [ -14.55934249],
       [ 748.71757498]]), array([[-110.84327485],
       [  94.90363393],
       [ 776.90506503]]), array([[-94.35306046],
       [ 46.62889867],
       [745.66214497]]), array([[ 12.27844431],
       [ 69.67892417],
       [674.59179271]]), array([[ -1.57925016],
       [ 60.53101544],
       [694.42792746]]), array([[104.17728066],
       [ 49.44370692],
       [699.70555399]])]

 vetores de rotação por imagem:

 [array([[-0.37627899],
        [-0.8451473 ],
        [-1.57734912]]), array([[-0.56800548],
        [-0.74686793],
        [-1.53628791]]), array([[-0.5060731 ],
        [-0.64746113],
        [-1.54207075]]), array([[-0.40753324],
        [-0.73637568],
        [-1.53721243]]), array([[-0.51404903],
        [-0.85142383],
        [-1.53144141]]), array([[-0.46851402],
        [-0.80549644],
        [-1.93600656]]), array([[-0.23555485],
        [-1.05168421],
        [-2.73917751]]), array([[0.06036876],
        [1.15384937],
        [2.87977384]]), array([[-0.39140906],
        [-1.12307555],
        [-2.85710671]])]
```

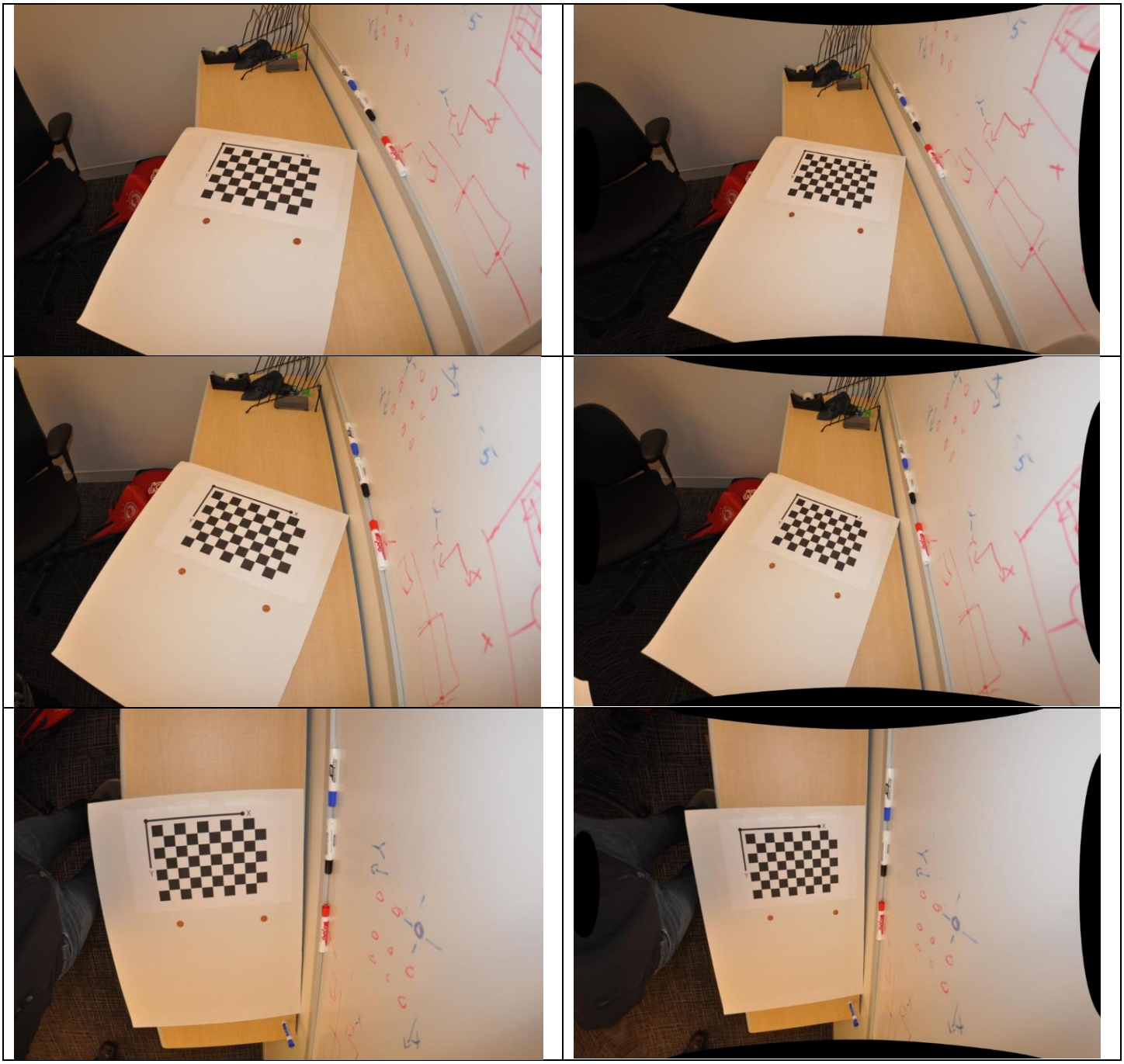
2. Dataset 1 “image_dist_X”

Assim como no dataset anterior, foi obtido todos os cantos dos tabuleiros das imagens utilizando os mesmos métodos. Para não ficar com uma abordagem repetitiva neste documento, as imagens estão na pasta “/results” com os nomes respectivos às imagens originais. Após a realização de todos os procedimentos para se encontrar a calibração de câmera, foi utilizada a função de python “cv2.undistort” para fazer a remoção da distorção das imagens deste conjunto. Ela tem como entrada a matriz de parâmetros intrínsecos e os coeficientes de distorção da lente obtidos anteriormente. Os resultados comparativos podem ser observados abaixo:



A partir da observação e comparação dos resultados, observa-se que as imagens recebidas possuem retas no mundo real curvas e paralelas no mundo real se interceptando. Porém após a remoção da distorção radial a partir do método de Zhang, as retas voltam a ser retas sem o aspecto curvo provocado pela lente da câmera. Este ajuste pode ser observado pelas curvas pretas nas bordas das imagens transformadas, proporcionando que o tabuleiro fique sem esta distorção. Porém observa-se que as bordas diagonais das imagens transformadas ficam bastante deformadas. Logo para uma utilização da imagem de forma mais eficiente é necessário fazer um corte dessas extremidades.