Homework1

Luu Van Duc Thieu

September 29, 2023

1 Exercise 1.1

a, I will divide the transformation scaling, rotating around point A(x, y) of point M(a, b) process into 4 phases:

1. Phase 1: Translate point M(a, b) to (c, d) so that the relative position of (c, d) to the O(0, 0) corresponds to the relative position of (a, b) to point A(x, y) using matrix A:

$$A = \begin{bmatrix} 1 & 0 & -x \\ 0 & 1 & -y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

2. Phase 2: Rotate M around O(0, 0) an angle θ using matrix B:

$$B = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

3. Phase 3: Isotrophic scale M with origin O(0, 0) s times using matrix C:

$$C = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

4. Phase 4: Translate M with vector (x, y) so that M will now keep the scaling and rotating status but with the origin being point A(x, y) using matrix D:

$$D = \begin{bmatrix} 1 & 0 & x \\ 0 & 1 & y \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

I combine both matrices in that order and transformation matrix of scaling, rotating around any point A(x, y) is:

$$T = D \cdot C \cdot B \cdot A = \begin{bmatrix} s \cdot \cos\theta & s \cdot \sin\theta & x - s \cdot x \cdot \cos\theta - s \cdot y \cdot \sin\theta \\ -s \cdot \sin\theta & s \cdot \cos\theta & y + s \cdot x \cdot \sin\theta - s \cdot y \cdot \cos\theta \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

b,

```python
def get_transform_matrix(rotate_angle, scale_factor, center_point):
  """
  Get the transformation matrix (scaling, rotate in center) to transform image
  Inputs:
    rotate_angle: float: rotation angle degree
    scale_factor: int: scale image
    center_points: list: [x, y]: co-ordinate of center point
  Returns:
    The numpy homogeneous transform matrix: (3x3)
  """
  x, y = center_point
  cos = np.cos(rotate_angle)
  sin = np.sin(rotate_angle)
  s = scale_factor
  normal = np.array([[s*cos, s*sin, x - s*x*cos - s*y*sin],
                     [-s*sin, s*cos, y + s*x*sin - s*y*cos],
                     [0, 0, 1]])
  return scale_factor * normal
```

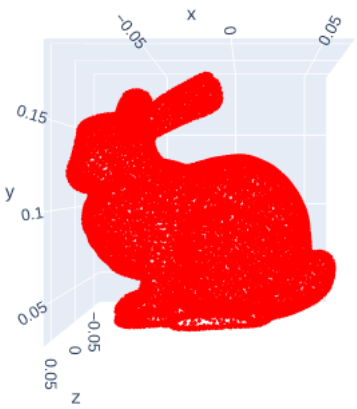
2 Exercise 1.2

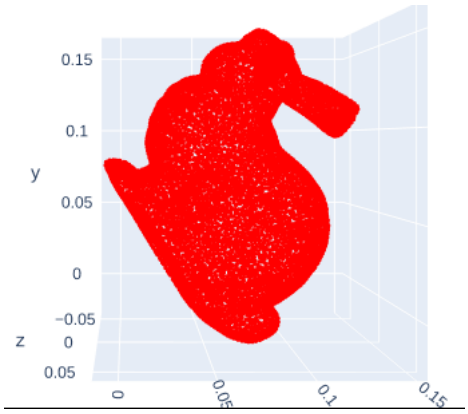
a, The form of homogeneous transformation in 3D of rotating α degree around Oz axis:

$$T = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

b, Demo:

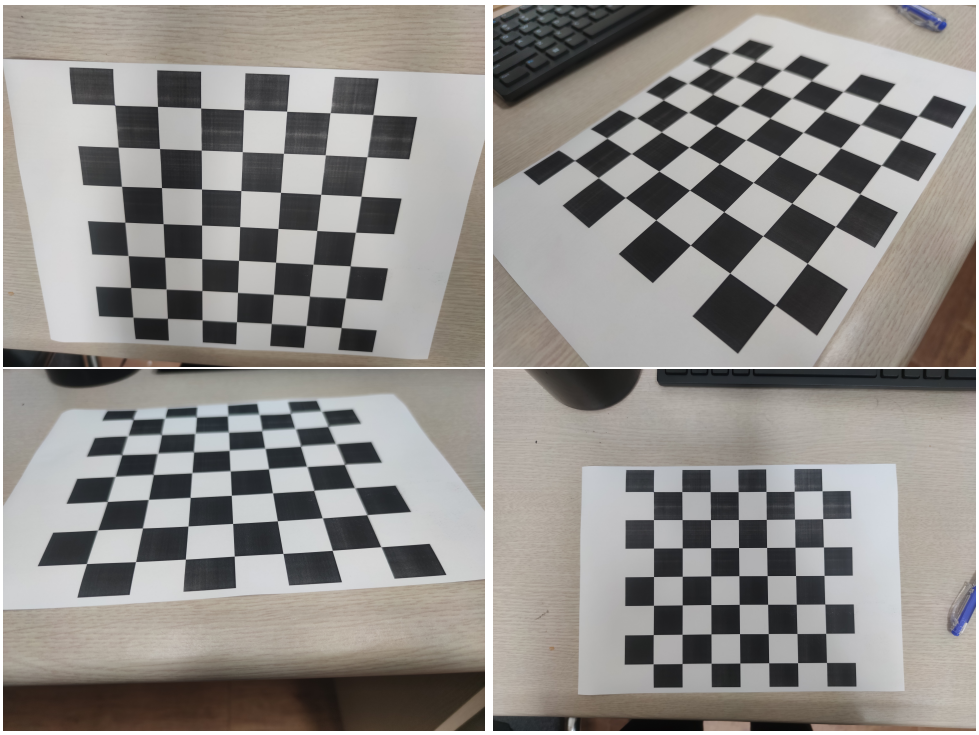
```python
def rotate_oz_3D_object(point_3d, rotate_angle):
  """
    Rotate object around Oz
    Inputs:
      point_3d: numpy array: shape (N, 3), N is number of points
      rotate_angle: float: rotate angle degree
    Outputs:
      rotated_point_3d: numpy array: shape (N, 3), N is number of points.
  To do:
    1. Create homogeneous rotated matrix (4x4)
    2. Convert point_3d into the homogeneous coordinate
    3. Matrix multiplication to get the result.
  """
  cos = np.cos(rotate_angle)
  sin = np.sin(rotate_angle)
  homogeneous = np.array([[cos, sin, 0, 0],
                          [-sin, cos, 0, 0],
                          [0, 0, 1, 0],
                          [0, 0, 0, 1]])
  ones = np.ones((point_3d.shape[0], 1))
  point_3d = np.concatenate([point_3d, ones], axis=1)
  point_3d = np.dot(homogeneous, point_3d.T)
  return point_3d.T[:, :-1]
```



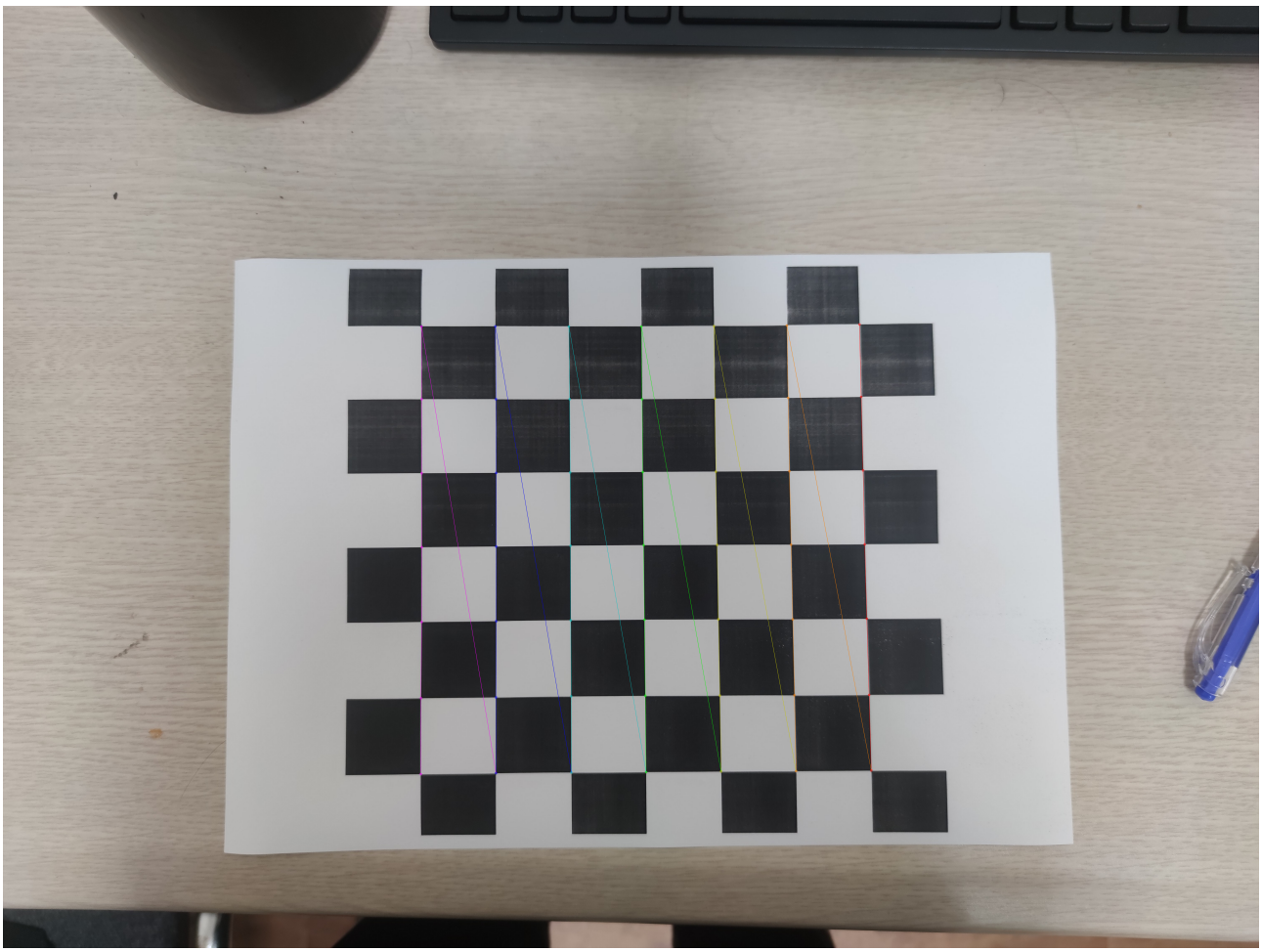


3 Exercise 2

1. Input images for the process:



2. Image containing the detected points:



3. Camera intrinsic matrix:

$$T = \begin{bmatrix} 15476.471210399912 & 0.0 & 3117.7589766048973 \\ 0.0 & 15824.91204000823 & 2199.136467822155 \\ 0.0 & 0.0 & 1.0 \end{bmatrix} \tag{7}$$