Exercise 2
3D Computer Vision

Shoaib Ahmed Siddiqui

01.12.2017

1 Theory

Properties of Rotation Matrices

1. Show that $U^T = U^{-1}$.

The most generic form of rotation matrices in 3D can be defined as:

$$R_z(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\psi) = \begin{bmatrix} cos(\psi) & 0 & sin(\psi) \\ 0 & 1 & 0 \\ -sin(\psi) & 0 & cos(\psi) \end{bmatrix}$$

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix}$$

Let us consider just one of the rotation matrix $R_z(\theta)$ along with its transpose:

$$R_z(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\theta)^T = \begin{bmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Given these two matrices, we can directly multiply them.

$$R_z(\theta) * R_z(\theta)^T = \begin{bmatrix} cos^2(\theta) + sin^2(\theta) & cos(\theta)sin(\theta) - sin(\theta)cos(\theta) & 0 \\ sin(\theta)cos(\theta) - cos(\theta)sin(\theta) & sin^2(\theta) + cos^2(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

$$\begin{aligned} sin^2(\theta) + cos^2(\theta) &= 1 \\ sin(\alpha - \beta) &= sin(\alpha)cos(\beta) - cos(\alpha)sin(\beta) \end{aligned} \tag{2}$$

Using the famous trigonometric identities given in Eq. 2, Eq. 1 can be written as:

$$R_z(\theta) * R_z(\theta)^T = \begin{bmatrix} 1 & sin(\theta - \theta) & 0 \\ sin(\theta - \theta) & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\theta) * R_z(\theta)^T = \begin{bmatrix} 1 & sin(0) & 0 \\ sin(0) & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\theta) * R_z(\theta)^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The same can be shown for $R_x(\psi)$ and $R_y(\phi)$.

2. Geometric interpretation of the determinent of a square 3×3 matrix A.
Matrix A performs a linear transformation of the input vector $\mathbf{x}$ as given in Eq 3.

$$A\mathbf{x} = \mathbf{b} \tag{3}$$

The determinent of matrix A represents the volume distortion experienced by the region defined by vector $\mathbf{x}$. The sign of the determinent represents if the transformation will preserve the orientation. Let us consider the input vector $\mathbf{x}$ to be of unit area in the standard basis. After the linear transformation through the matrix A, the vector $\mathbf{x}$ will change its area to $|(detA)|$. Since for rotations, translations, and other linear transformations, we would like to preserve the scale of the input, therefore, we take matrices with unit determinent. This leads to input transformation without any change in scale [1].

Transformation Chain
The step step in the transformation chain is to transform the point from the world coordinate system to the camera coordinate system. We consider two types of transformations between the coordinate frames. These two transformations are rotation and translation. Let us consider a point in 3D defined by $\mathbf{x}$ in the world coordinate system. Then, the projection of $\mathbf{x}$ in the camera coordinate system can be denoted via $\tilde{\mathbf{x}}$ and can be computed as:

$$\tilde{\mathbf{x}} = R\mathbf{x} + \mathbf{t} \tag{4}$$

In Eq. 4, R is the rotation matrix while $\mathbf{t}$ is the translation vector. The projection from 3D to 2D can be achieved by throwing out the last coordinate, i.e. $(x, y, z) \rightarrow (f'x/z, f'y/z)$. Division by z makes this transformation non-linear. This transformation can be converted to a linear transformation via adding another coordinate to the vector making it homogeneous i.e. $(x, y, w) \rightarrow (x/w, y/w)$ for 2D points and $(x, y, z, w) \rightarrow (x/w, y/w, z/w)$ for 3D points. The final coordinate represents the scale, hence, homogeneous coordinates are scale invariant. Hence, the transformation from world coordinate system to camera coordinate system can be rewritten as:

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ 1 \end{bmatrix} = \begin{bmatrix} R & \mathbf{t} \\ 0_3^T & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

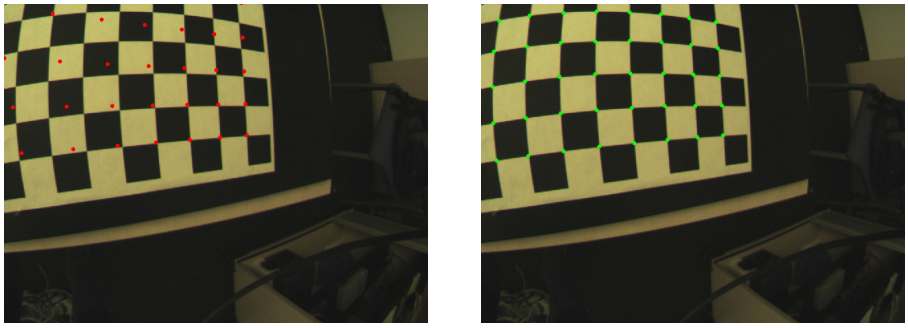
In this linear transformation, R and T are extrinsic parameters of the camera. The point $\tilde{\mathbf{x}}$ in camera coordinate system has to now go through the perspective transformation and final projection onto the image plane. This transformation is captured by the intrinsic camera parameters K. Let α_x, α_y be the focal length of the camera in pixels, (x_0, y_0) be the coordinates of the image center in pixels and s be the skew parameter, then the final transformation mapping the camera coordinates to image plane can be written as:

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} \alpha_x & s & x_0 & 0 \\ 0 & \alpha_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ 1 \end{bmatrix}$$

The 2D homogeneous coordinates can be converted to 2D point via normalization of the scale i.e. $(u', v', w') \rightarrow (u'/w', v'/w')$. The normalized point $(u'/w', v'/w')$ can be denoted as (x_{pix}, y_{pix}) specifies the location of the point in the image plane.

2 Implementation

Fig. 1 visualizes the output of the main.py script which transforms the points provided in the 3D world coordinate system to the image coordinates. Fig. 1 (left) is the output without correction of radial distortion present in the camera. It is evident from the image that the visualized points are not perfectly aligned with the corners present in the checkerboard. Once the radial distortion is catered through the provided distortion parameters, the points got perfectly aligned with the corners in the checkerboard which is visualized in Fig. 1 (right).



(a) (b)

Figure 1: (a) Without correction of radial distortion. (b) With correction of radial distortion.

References

[1] Determinants and linear transformations. http://mathinsight.org/determinant_linear_transformation.